

# Branch-and-cut and Branch-and-Cut-and-Price Algorithms for Solving Vehicle Routing Problems

**Jepsen, Mads Kehlet**

*Publication date:*
2011

*Document Version*
Publisher's PDF, also known as Version of record

Technical University of Denmark

DTU

PHD Thesis

# Branch-and-cut and Branch-and-Cut-and-Price Algorithms for Solving Vehicle Routing Problems

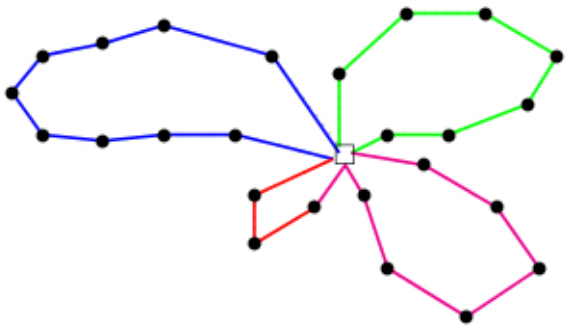
*Author:* Mads Kehlet Jepsen

*Supervisor:* Professor David Pisinger

$$\min \sum_{e \in E} c_e x_e$$
$$\text{s.t.} \sum_{e \in \delta(i)} x_e = 2 \qquad \forall i \in V_c$$
$$\sum_{e \in \delta(0)} x_e = 2|K|$$
$$\sum_{e \in \delta(S)} x_e \geq 2r(S) \qquad \forall S \subseteq V_c, |S| \geq 2$$
$$x_e \in \{0,1,2\} \qquad \forall e \in \delta(0)$$
$$x_e \in \{0,1\} \qquad \forall e \in E \setminus \delta(0)$$

$$\min \sum_{p \in P} c_p \lambda_p$$
$$\text{s.t} \sum_{p \in P} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp} \lambda_p = 1 \qquad \forall i \in V_c$$
$$\lambda_p \in \{0,1\} \qquad \forall p \in P$$

$$\min \sum_{e \in E} c_e x_e - \sum_{i \in N} p_i y_i$$
$$\sum_{e \in \delta(i)} x_e = 2y_i \qquad \forall i \in V$$
$$y_0 = 1$$
$$\sum_{e \in \delta(S)} x_e \geq 2y_i \qquad \forall i \in S, \forall S \subseteq V_c, |S| \geq 2$$
$$\sum_{i \in N} d_i y_i \leq Q$$
$$x_e \in \{0,1\} \qquad \forall e \in E$$
$$y_i \in \{0,1\} \qquad \forall i \in V.$$

8. Juli 2011

# Preface

This Ph.D thesis was started at the Department of Computer Science, University of Copenhagen (DIKU) in collaboration with Microsoft Development Center Copenhagen (MDCC). It was continued and finished at the Department of Management Engineering, Danish Technical University (DTU Man) at the OR-section. I would like to thank the University of Copenhagen, Microsoft Development Center Copenhagen and the Danish Technical University for granting me the opportunity to study a Ph.D.

I would like to thank my old colleagues at DIKU and MDCC and my colleagues at DTU Man for creating an excellent working environment during the past three year. I would like to thank Berit Løfsted, Christian E. M. Plum, Line Blander Reinhardt, Stefan Røpke and Mikkel M. Sigurd for the interesting discussion during the creation of the articles I have written with you. I would especially like to thank Bjørn Petersen, Simon Spoorendonk and my supervisor David Pisinger for the many discussions, we have had during the construction of our papers. Finally I like to thank my wife Line for a lot of proof reading throughout the entire thesis. Thanks

# Contents

# Chapter 1

# Introduction

The Vehicle Routing Problem was introduced by Dantzig and Ramser [19] under the name the truck dispatching problem. The problem consists of finding $m$ routes covering $n$ customers with a common point called the depot. The goal of the optimization is to find the set of routes with the minimal overall distance. As time has passed, many variants of the problem have been proposed. Today there probably exist more than 250 different variants and at every major conference several new variants are proposed. The main reason for this rapid growth in the number of problems are, in my opinion be attributed to the importance of the problems in the real world and the excellent solution methods that have been developed. For exact solution methods the two problems that have been the center of the attention is the Capacitated Vehicle Routing Problem (CVRP) where each costumer has a demand and the vehicle has a fixed capacity, and the Vehicle Routing Problem with Time Windows (VRPTW) which is a CVRP problem where each customer can only be serviced within a predefined time window.

Before 1980 very few exact algorithms for CVRP and VRPTW had been proposed, but in the early 1980s two new exact methods where proposed. From this point the history of exact methods for CVRP and VRPTW can be divided into three phases. The first phase was the introduction of the Set Partition and the development of Branch-and-Cut-and-Price (BP) algorithms using a relaxed pricing problem. The second was the development of Branch-and-Cut (BAC) algorithms. In the current phase the pricing problem is no longer relaxed and cuts in the master problem of the Branch-and-Cut-and-Price algorithms is used. The first two phases where started at the same point in time and there is still development on the algorithms in the context of CVRP and VRPTW. The algorithms from these two phases are also used on several other variants of the Vehicle Routing Problem. The third phase was started in the middle of the 2000s and the algorithms from this phase are currently the best overall performing algorithms.

The main idea of this chapter is to review some of the known concepts

and methods from the literature. Whenever a method or concept is central for the thesis it will be explained in more detail, while other concepts or methods are only briefly mentioned. Section 1.1 is dedicated the CVRP and VRPTW, while section 1.2 is dedicated the Capacitated Location Routing Problem. For the CVRP and VRPTW section 1.1.1 will review the basic Branch-and-Cut algorithms and state two of the cutting planes which has been of great importance for this thesis. Section 1.1.3 will review the Branch-and-Cut-and-Price algorithms for the CVRP and VRPTW. The section will contain information about how cuts are handled and it will review the solution methods for the Resource Constraint Shortest Path Problem and Elementary Resource Constraint Shortest Path Problem, which are some of the possible pricing problems. Either the BAC and BCP principles for the CVRP and VRPTW or both serves as inspiration in all of the academic papers in Part I and Part II. Section 1.2 will introduce some mathematical models for the Capacitated Location Routing Problem and some of the fundamental cutting for this problem. Both the models and cutting planes have served as inspiration for the solution approach for the article in Chapter 3. Finally section 1.3 will give a short summary of the academic papers in the thesis.

## 1.1 CVRP and VRPTW

The symmetric capacitated vehicle routing problem can be formulated as follows. Let $V$ be a vertex set that consists of a set of customers $V_c$ and the depot set $V_0 = \{0\}$ and $K$ be a set of vehicles. With each node $i \in V_c$ there is associated a positive integer $d_i$, which we refer to as the demand. Let $E$ be the set of edges connecting the vertices and let $c_e, e \in E$ be the cost of traversing the edge. A feasible solution to CVRP is $|K|$ cycles starting in the depot, each customer is visited once and the sum of the customers visited on the cycles does not exceed the capacity $C$. In the optimization version we are seeking the feasible solution where the sum of the edge weights of the selected cycles is minimal. CVRP is often defined on a complete graph $G(V, E)$ and test instances are available at *www.branchandcut.org*. Rather than referring to a solution as a set of cycles the abbreviation routes is often used and we will therefore use this.

A natural extension of CVRP is the inclusion of time windows. A time window is defined as a time interval $[a_i, b_i]$ for which a customer $i \in V_c$ can be serviced by the vehicle. Furthermore, a service time $s_i$ is associated with the customer $i \in V_c$. To include the time windows an asymmetric formulation is often used. Let $A$ be a set of arcs, where each arc $(i, j) \in A$ has a cost $c_{ij}$ and a travel time $\tau_{ij}$ associated. For the depot the lower value $a_0 = 0$ and the upper value $b_0 = T$. Given a route $R$ with the ordered vertex set $V(R) = \{v_0, v_1, v_2, ..., v_{k-1}, v_k\}$ where $v_0 = v_k = 0$ the accumulated time for

the route at a given customer is recursively defined as:

$$T(i) = \begin{cases} 0 & i = 0 \\ \max\{T(v_{i-1}) + \tau_{v_{i-1},v_i} + s_{v_{i-1}}, a_{v_i}\} & i \neq 0 \end{cases}$$

A route $R$ is said to be feasible if $T(i) \leq b_i, \forall i \in V(R)$. It is possible to include the service time $s$ in the travel time $\tau_{ij}$ by setting $\tau_{ij} = \tau_{ij} + s_j$.

Aside from the feasibility of a route, VRPTW is similar to CVRP, with the only difference being that the number of routes in the minimum cost solution is not predefined. The time windows are often tightened using the rules suggested by Desrochers et al. [20] where the earliest arrival and latest departure times are used to reduce the width of the time windows. Furthermore, arc elimination can be carried out using a series of shortest path calculations. Traditionally new algorithms for the VRPTW is tested on the 56 instances introduced by Solomon [60]. These benchmarks are naturally refered to as the Solomon instances.

Throughout the thesis, some shorthand notations will be used when modelling various VRPs. In directed problems the two terms $\delta^-(S)$ and $\delta^+(S)$ for some subset $S \subseteq V$ are define as:

- $\delta^-(S) = \{(i,j) : i \in V \setminus S, j \in S, (i,j) \in A\}$, that is the set of arcs entering the set $S$.

- $\delta^+(S) = \{(i,j) : i \in S, j \in V \setminus S, (i,j) \in A\}$, that is the set of arcs leaving the set $S$.

Similar in the undirected variants, the term $\delta(S)$ for some $S \subseteq V$ refers to the possible empty set of edges with an endpoint in $S$ and one in $V \setminus S$. Common for both undirected and directed problems is the use of $E(S)$ as the arcs/edges where both end points are in $S$ and $E(S_1 : S_2)$ which is the set of arcs/edges which originates in $S_1$ and ends in $S_2$.

One of many mathematical formulations of VRPTW which also includes CVRP is the three-index flow model (see Toth and Vigo [62]). The model is based on the model introduced by Fisher and Jaikumar [30]. Let respectively $\{o\}$ and $\{o'\}$ denote the depot at the start and end of the route. Assume that the number of vehicles $|K|$ is unbounded. Let $t_{ik}$ be the time vehicle $k \in K$ visits node $i \in V$, if the vehicle visit the node and undefined otherwise. Let $x_{ijk}$ be a binary variable indicating whether vehicle $k \in K$ traverses arc $(i,j) \in A$. A mathematical model for CVRP and VRPTW can then be

formulated as:

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ijk} \tag{1.1}$$

$$\text{s.t.} \sum_{k \in K} \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 1 \qquad \forall i \in V_c \tag{1.2}$$

$$\sum_{(i,j) \in \delta^+(o)} x_{ijk} = \sum_{(i,j) \in \delta^-(o')} x_{ijk} = 1 \qquad \forall k \in K \tag{1.3}$$

$$\sum_{(j,i) \in \delta^-(i)} x_{jik} - \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 0 \qquad \forall i \in V_c,\ \forall k \in K \tag{1.4}$$

$$\sum_{(i,j) \in E} d_i x_{ijk} \leq C \qquad k \in K \tag{1.5}$$

$$a_i \leq t_{ik} \leq b_i \qquad \forall i \in V,\ \forall k \in K \tag{1.6}$$

$$x_{ijk}(t_{ik} + \tau_{ij}) \leq t_{jk} \qquad \forall (i,j) \in A,\ \forall k \in K \tag{1.7}$$

$$x_{ijk} \in \{0,1\} \qquad \forall (i,j) \in A,\ \forall k \in K \tag{1.8}$$

Constraints (1.2) ensures that every customer $i \in V_c$ is visited, while constraints (1.3) ensures that each route starts and ends in the depot. Constraint (1.4) maintains flow conservation, while (1.5) ensures that the capacity of each vehicle is not exceeded. Constraints (1.6) and (1.7) ensure that the time windows are satisfied. Note that (1.7) together with the assumption that $\tau_{ij} > 0$ for all $(i,j) \in E$ eliminates sub-tours. The last constraints define the domain of the arc flow variables. Note that a zero-cost edge $x_{oo'k}$ between the start and end depot must be present for all vehicles for (1.3) to hold if not all vehicles are used. Constraints (1.7) are non linear but can easily be linearized if needed. If the value of the travel time $\tau_{ij}$ is set to 1 and $a_i = 0$ and $b_i = |V| + 1$ the model is a formulation of the CVRP. In this case the constraints correspond to the MTZ constraint introduced for the traveling salesman problem by Miller et al. [49].

### 1.1.1 Branch-and-Cut

Most Branch-and-Cut algorithms for VRP are based on the two index formulation introduced by Laporte and Nobert [42]. Let $x_e, e \in E \setminus \delta(V_0)$ be a binary variable which is 1 iff the edge is used in the solution. Let $x_e, e \in \delta(V_0)$ be an integer variable which is 1 iff the edge is used once and 2 iff the edge is used twice. When an edge out of the depot is used twice it corresponds to a route which visits only a single customer. For some $S \subseteq V_c$ let $r(S)$ be a lower bound on the number of vehicles needed to service $S$.

The mathematical model by Laporte and Nobert [42] can then be stated as:

$$\min \sum_{e \in E} c_e x_e \tag{1.9}$$

$$\text{s.t.} \sum_{e \in \delta(i)} x_e = 2 \qquad \forall i \in V_c \tag{1.10}$$

$$\sum_{e \in \delta(0)} x_e = 2|K| \tag{1.11}$$

$$\sum_{e \in \delta(S)} x_e \geq 2r(S) \qquad \forall S \subseteq V_c, |S| \geq 2 \tag{1.12}$$

$$x_e \in \{0, 1, 2\} \qquad \forall e \in \delta(0) \tag{1.13}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E \setminus \delta(0) \tag{1.14}$$

The objective minimizes the cost of the selected routes, constraints (1.10) and (1.11) ensure that each customer is visited once and that $|K|$ vehicles are used. Constraints (1.12) are the capacity constraints which ensure that the number of vehicles servicing the set $S$ is sufficient. Finally (1.13) and (1.14) define the domains of the variables. Cornuejols and Harche [17] have shown that it is sufficient to use $r(S) = \left\lfloor \frac{\sum_{i \in S} d_i}{C} \right\rfloor$ for the model to be a correct formulation of the CVRP.

The basic idea in a Branch-and-Cut algorithm is to solve a Linear relaxation of the corresponding Integer Programming Problem and then cut and branch when possible and needed. In the case of CVRP this means that the domains of the variables (1.13) to (1.14) are substituted with the linear bounds:

$$0 \leq x_e \leq 2 \qquad \forall e \in \delta(0) \tag{1.15}$$

$$0 \leq x_e \leq 1 \qquad \forall e \in E \setminus \delta(0) \tag{1.16}$$

Since the number of constraints (1.12) is exponential, the running time of solving the linear relaxation is exponential, therefore these are removed and added when needed. To identify when one of the inequalities (1.12) is violated a separation problem is solved. Let $x^*$ be a solution to the linear model (1.9) to (1.11), with the bounds (1.15) and (1.16). Any off the constraints (1.12) is said to be violated if:

$$\sum_{e \in \delta(S)} x_e^* < 2r(S), \qquad S \subset V_c, |S| \geq 2$$

And the separation problem is to find a violated inequality. When such an inequality is identified it is referred to as a cut since it cuts off the current linear solution. Naddef and Rinaldi [51] have shown that the separation of inequalities (1.12) is NP-hard and Lysgaard et al. [48] have developed several heuristics to find violated inequalities for the rounded version.

In a BAC algorithm cuts are added iteratively and once no more cuts can be found branching is performed. Naddef and Rinaldi [51] have suggested to branch on a set $S$ where $2 < x^*(S) < 4$. The branch is then imposed with the two constraints $x(\delta(S)) = 2$ and $x(\delta(S)) \geq 4$. When $|S| = 2$ this corresponds to an edge branch, since $x(\delta(S)) = 2 - 2x(E(S))$ and $E(S)$ only contains the edge between the two nodes. Therefore an alternative formulation of the two branches is $x(E(S)) = 1$ and $x(E(S)) \leq 0$, which forces the edge to be either 0 or 1.

Lysgaard et al. [48] have developed a very successful BAC algorithm and have furthermore published the source code for the separation algorithms in the package CVRPSEP [47]. The separation routines have been used in many of this thesis academic papers, but the methods have not been altered and we therefore refer the reader to Lysgaard et al. [48] for details. Instead we will focus on two valid inequalities that are used as a basis for many of the inequalities in the academic papers in chapter 3 and 4. The Generalized Large Multistar inequalities(GLM) were introduced by Letchford and Salazar-González [46] and can be stated as:

$$\sum_{e \in \delta(S)} x_e \geq \frac{2}{C} \left( \sum_{i \in S} d_i + \sum_{j \in V_c \setminus S} \sum_{e \in E(j:S)} d_j x_e \right) \; S \subset V_c, |S| \geq 2 \qquad (1.17)$$

The idea in the GLM is to consider the number of vehicles needed to service the set $S$. A valid lower bound for this is the demand of the set plus any customer visited directly from the set divided by the capacity. The GLM can be separated in polynomial time by solving a series of minimum cut problems. The Knapsack Large Multistar(KLM) inequality introduced by Letchford et al. [44] and Letchford and Salazar-González [45] show how to form these inequalities based on the three-index flow model and then project them to the two index formulation. The knapsack polytope is defined as:

$$P_K = \text{conv} \left\{ y \in \{0,1\}^{|V_c|} : \sum_{i \in V_c} d_i y_i \leq C \right\}$$

Let $a, b \geq 0$ and let the inequality $\sum_{i \in V_c} a_i y_i \leq b$ be valid for $P_K$. Then the KLM is defined as:

$$\sum_{e \in \delta(S)} x_e \geq \frac{2}{b} \left( \sum_{i \in S} a_i + \sum_{j \in V_c \setminus S} \sum_{e \in E(j:S)} a_j x_e \right) \qquad S \subseteq V_c, |S| \geq 2 \qquad (1.18)$$

There are many ways to select the coefficients $a$ and $b$. Letchford et al. [44] suggest to find a violated cover inequality and then use the polynomial time algorithm for the GLM inequalities to separate the most violated KLM inequality.

BAC algorithms for the CVRP have been very successful, but for the VRPTW the results are not as convincing although some good results exist. Bard et al. [6] were among the first to develop a BAC algorithm for the VRPTW. They present a compact two index flow formulation which uses $2|V_c|$ continuous variables and $|A|$ binary variables. The objective considered were to minimize the number of vehicles, which is slightly different from minimizing the cost of the selected routes, however this is easily changed in their model. For all $(i,j) \in A$ let $x_{ij}$ be a binary variable, which is one if the arc is used. For all $i \in V_c$ let $y_i$ be the vehicles load when departing the customer and $t_i$ equal the time when the vehicle departs customer. We assume that $t_0 = 0$. Let $M_{ij} = b_i - a_j$, a mathematical model for the VRPTW can then be formulated as:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{1.19}$$

$$\text{s.t.} \sum_{(ij)\in\delta^+(i)} x_{ij} = 1 \qquad \forall i \in V_c \tag{1.20}$$

$$\sum_{(i,j)\in\delta^+(i)} x_{ij} = \sum_{(j,i)\in\delta^-(i)} x_{ji} \qquad \forall i \in V_c \tag{1.21}$$

$$t_j \geq t_i + \tau_{ij}x_{ij} - M_{ij}(1 - x_{ij}) \qquad \forall i \in V, \forall j \in V_c \tag{1.22}$$

$$y_j \geq y_i + q_j - C(1 - x_{ij}) \qquad \forall i,j \in V_c^2 \tag{1.23}$$

$$a_i \leq t_i \leq b_i \qquad \forall i \in V_c \tag{1.24}$$

$$d_i \leq y_i \leq C \qquad \forall i \in V_c \tag{1.25}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{1.26}$$

$$\tag{1.27}$$

The objective (1.19) minimizes the cost of the selected arcs and thereby the cost of the routes. Constraints (1.20) and (1.21) ensure that a customer is visited once and that a customer have both an entering and leaving arc. Constraints (1.22) and (1.23) ensure that the time and capacity when the vehicle leaves is set correctly. (1.22) can be explained as follows: When the arc between customers $i$ and $j$ is not used($x_{ij} = 0$) the constraints reduces to:

$$t_j \geq t_i - b_i + a_j \qquad \forall i \in V, \forall j \in V_c$$

From the domain constraints (1.24) it follows that $t_i \leq b_i$ and therefore $t_i - b_i \leq 0$, which implies that $t_j \geq a_j$ which is correct and does not cut off any feasible solution. In the case where the arc is used($x_{ij} = 1$) it follows that:

$$t_j \geq t_i + \tau_{ij} \qquad \forall i \in V, \forall j \in V_c$$

This correspond to accumulating the time of the route. A similar analysis can be made for constraints (1.23). The domains of the variables are defined by constraints (1.24) to (1.26).

Although (1.22) and (1.23) can be used to obtain an integer solution it is possible to omit these constraints and replace them with an exponential set of constraints. Kallehauge et al. [37] followed this direction and replaced constraints (1.22) and (1.23) with path inequalities. A path $Q$ is an ordered vertex set with vertices $V(Q) = \{v_1, v_2, ..., v_{k-1}, v_k\}$ and an arc set $A(Q)$. A path is infeasible if either $\sum_{i \in V(Q)} d_i > C$ or $T(v_i) > b_{v_i}$ for some $v_i \in V(Q)$. If the path $Q$ is infeasible the inequality:

$$x(A(Q)) \leq |A(Q)| - 1$$

is a valid inequality. If $\mathcal{Q}_I$ is the set of all infeasible paths then it is possible to substitute the constraints (1.22) to (1.25) with:

$$x(A(Q)) \leq |A(Q)| - 1 \qquad \forall Q \in \mathcal{Q}_I \tag{1.28}$$

It is possible to strengthen the infeasible path inequalities and show that these are facet defining(see Kallehauge et al. [37] for details). The corresponding BAC algorithm based on the model with infeasible path inequalities resulted in the solution of a previously unsolved Solomon instance. Kallehauge et al. [37] also suggest to include precedence constraints and Bard et al. [6] showed that many of the inequalities known from the Traveling Salesman polytope can be used.

### 1.1.2 Set Partition Formulation

The set partition formulation was introduced by Balinski and Quandt [5]. Let $P$ be the set of all feasible routes, the binary constant $\alpha_{ijp}$ is 1 iff arc $(i, j)$ is used by route $p \in P$, and the binary variable $\lambda_p$ indicates whether route $p$ is used. The Set Partitioning formulation of VRP is then:

$$\min \sum_{p \in P} \sum_{(i,j) \in A} c_{ij} \alpha_{ijp} \lambda_p \tag{1.29}$$

$$\text{s.t} \sum_{p \in P} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp} \lambda_p = 1 \qquad \forall i \in V_c \tag{1.30}$$

$$\lambda_p \in \{0, 1\} \qquad \forall p \in P \tag{1.31}$$

The cost function (1.29) minimized the cost of the selected routes, constraints (1.30) ensure that all customers are visited once and constraints (1.31) are the domains of the variables. The Set Partition formulation is not usable in practice since the number of feasible routes is exponential in worst case. It is, however usable in a column generation approach which we shall consider in the next section.

### 1.1.3 Branch-and-Cut-and-Price

Branch-and-Cut-and-Price(BCP) algorithms originate from solving the Set Partition model of VRP. The Set Partition formulation can be viewed as a Danzig-Wolfe reformulation of the three-index-flow formulation where the sub problems are recognized to be identical. In a BCP algorithm a restricted master problem containing a partial portion of the columns is solved to linear optimality and from that solution, new columns and cuts are generated.

The binary constraints on the variables are relaxed to $0 \leq \lambda_p \leq 1, \forall p \in P$ and the routes are generated on demand using the so called pricing problem. The pricing problem considers the reduced cost of a column by taking the current dual solution into account. Consider a subset of columns $\mathcal{P} \subseteq P$. The corresponding reduced and relaxed master problem is then:

$$\min \sum_{p \in \mathcal{P}} \sum_{(i,j) \in A} c_{ij} \alpha_{ijp} \lambda_p \tag{1.32}$$

$$\text{s.t} \sum_{p \in \mathcal{P}} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp} \lambda_p = 1 \qquad \forall i \in V_c \tag{1.33}$$

$$0 \leq \lambda_p \leq 1 \qquad \forall p \in \mathcal{P} \tag{1.34}$$

Let $\pi \in \mathbb{R}$ be the dual variables of (1.33) and let $\pi_0 = 0$. The reduced cost of a route in $p \in P \setminus \mathcal{P}$ is then given as:

$$\overline{c}_p = \sum_{(i,j) \in E} c_{ij} \alpha_{ijp} - \sum_{(i,j) \in E} \pi_j \alpha_{ijp} = \sum_{(i,j) \in E} (c_{ij} - \pi_j) \alpha_{ijp} \tag{1.35}$$

The pricing problem is then to solve an Elementary Resource Constrained Shortest Path Problem with Resource Constraints(ESPPRC). Given a directed weighted graph $G(V, E)$ with weight function $w(u, v), (u, v) \in E$ and a resource set $R$, where the resources $R$ are a time window resource and a capacity resource, the ESPPRC is defined as follows: Find the shortest path from a source node $s$ to a target node $t$ such that the path is feasible with respect to the resources. To solve the pricing problem for VRP as an ESPPRC, the weights on the edges is set as:

$$w(i,j) = c_{ij} - \frac{1}{2}\pi_i - \frac{1}{2}\pi_j, \quad (i,j) \in A$$

The source and the target are set to be the depot out and the depot in. Note that subtracting half the dual from cost will help maintain symmetry if the original cost is also symmetric which is the case in CVRP. When solving the specific pricing problems of CVRP and VRPTW the problem solved is referred to as respectively the Elementary Shortest Path Problem with Capacity Constraint(ESPPCC) and Elementary Shortest Path Problem with Capacity Constraint and Time Windows(ESPPTWCC). Dror [25] has proven that the ESPPTWCC is NP-hard. The reduction is based on the time windows so it

does not apply to the ESPPCC case. However the reduction for this ESPPCC can easily be made from the Traveling Salesman Problem and will only rely on the fact that the graph may contain negative cycles.

Rather than solving the master problem using a linear optimization solver Christofides et al. [13] suggested to find the dual variables through Lagrangian relaxation. The same approach has been followed by Kohl and Madsen [38] and Baldacci et al. [3] for VRPTW and Baldacci et al. [1] for CVRP who use subgradient optimization until the last stages of the algorithms. Solving the Lagrangian dual has also been explored by Kallehauge et al. [36] who use a trust region to stabilize the duals. Furthermore the work by Kolen et al. [40] can also be viewed as an Lagrangian approach.

### Solving the Pricing Problem

In early column generation approaches from the late sixties to the late seventies the algorithms reduced the route sets, to routes with a given mathematical structure. This includes the algorithms proposed by Rao and Zionts [56] and Foster and Ryan [31]. The reduction in the size of the Set Partition problem and the reduction in the complexity of the pricing problem made it possible to obtain feasible solutions.

In the early eighties an alternative approach, where the elementary property of the routes where relaxed, was proposed. This includes the so-called $q$-route relaxation suggested by Christofides et al. [13] based on the state-space relaxations by Christofides et al. [14] and the SPTW (Shortest Path With Time Windows) relaxation suggested by Desrosiers et al. [24]. The results of these relaxed problems are a connected path where a customers may be visited more than once. In both cases path containing cycles of size two were eliminated. In general this problem is referred to as the 2-cycle free shortest path problem (2SPPRC) . A relaxation of the route set where 2SPPRC is used provide a valid lower bound to the VRPs and by branching on edges/arcs in the original problem (variables $x_{ijk}$) a feasible integer solution can be obtained.

In both of the above papers the solution method for the 2SPPRC was dynamic programming. A state $L$ in the dynamic programming table is often referred to as a label. A label represents a partial solution which is feasible with respect to the resources. Each label is connected to a node in the graph and the labels are constructed by extending some label from its current node to a new node in the graph. Each label has the following functions associated:

- $c(L)$ is the cost of the label.
- $v(L)$ is the node the label is connected to.
- $\Pi(L)$ is the predecessor label, from which the label was constructed.

- $d(L)$ is the accumulated capacity of the label and is defined recursively as $d(L) = d(\Pi(L)) + d_{v(L)}$.
- $t(L)$ is the accumulated time of the label and is defined recursively as:$t(L) = \max\{a_{v(L)}, t(\Pi(L)) + \tau_{v(\Pi(L)),v(L)}\}$.

A label represents a feasible path segment from the depot to the current node $v(L)$. A feasible extension to the label is therefore a path segment from the labels current node $v(L)$ to the target node, such that the two combined segments are feasible. In the case of 2SPPRC this excludes extensions which visit the node $v(\Pi(L))$ as the first node. However, an extension which visit the node as the second node is feasible.

To avoid labels that cannot lead to an optimal solution, the concept of dominance has been introduced.

**Definition 1.** *A label $L$ is dominated if there exist no feasible extension that can lead to an optimal solution.*

A very simple dominance rule which is:

**Dominance 1.** *A label $L_1$ dominates a label $L_2$ if:*

$$c(L_1) \leq c(L_2) \quad d(L_1) \leq d(L_2) \quad t(L_1) \leq t(L_2) \quad v(\Pi(L_1)) = v(\Pi(L_2))$$

The idea behind dominance rule 1 is that any feasible extension of the label $L_2$ is also feasible for the label $L_1$. Since the cost of $L_1$ is less than the cost of $L_2$ we can conclude that a feasible solution constructed using the label $L_1$ will always be less expensive.

A slightly more complex dominance rule is to use two labels to dominate a single label:

**Dominance 2.** *A label $L_1$ and $L_2$ dominate a label $L_3$ if $v(\Pi(L_1)) \neq v(\Pi(L_2))$ and:*

$$\begin{aligned} c(L_1) \leq c(L_3) \quad d(L_1) \leq d(L_3) \quad t(L_1) \leq t(L_3) \\ c(L_2) \leq c(L_3) \quad d(L_2) \leq d(L_3) \quad t(L_1) \leq t(L_3) \end{aligned}$$

The idea in dominance rule 2 is that the only feasible extension of label $L_3$ that label $L_1$ does not cover are the ones that start in the node $v(\Pi(L_1))$. Since $v(\Pi(L_1)) \neq v(\Pi(L_2))$ these extensions are feasible for label $L_2$.

The above two propositions form the basis for the pseudo polynomial algorithms proposed by the Christofides et al. [14] and Desrosiers et al. [24] for the two special cases of 2SPPRC . The algorithm can be implemented in a general labeling framework for solving resource constrained shortest path problems. The generic framework is given in algorithm 1. As input the algorithm takes the data of the instance and returns a set of labels which can be converted to a feasible solution. The algorithm uses a priority queue

(see [16]) and two auxiliary functions. $EXTENDLABEL$ constructs a new label $NewL$ from a label $L$ and ensures that both the capacity and the time does not violate any of the rules. The function $DOMINATE$ tests the appropriate dominance rules for the concrete algorithm and updates the set of non dominated labels in the node $v$ ($\mathcal{L}_v$). Lines 1-3 construct an initial label at the start node, enqueues the label in the priority queue and initializes the solution pool with the empty set. As long as there are unprocessed labels a label is selected in lines 5-6. Lines 8-11 try to create a new label from the selected label for each node adjacent to the labels node. Lines 12 to 14 store the new label if it was an extension to the target node. Line 15-18 call the dominance function and store the new label if needed. Finally all solutions are returned in line 21. To solve the 2SPPRC the function $DOMINATE$ is

**Algorithm 1** Generic label algorithm

```
1: GENSPPRC(G, s, t, d, C, a, b, τ, T)
2: L ← {0, s, 0, 0, 0}
3: PQ.ENQUEUE(L)
4: SOL ← ∅
5: while PQ.TOP() ≠ ∅ do
6:    L ← PQ.DEQUEUE()
7:    for e(u, v) ∈ δ⁺(v(L)) do
8:       NewL ← EXTENDLABEL(L, v, d, C, a, b, τ, T)
9:       if NewL = NIL then
10:         continue
11:      end if
12:      if v = t then
13:         SOL.ENQUEUE(Newl)
14:      end if
15:      if DOMINATE(NewL, L_v) then
16:         continue
17:      end if
18:      PQ.ENQUEUE(NewL)
19:   end for
20: end while
21: return SOL
```

implemented using proposition 1 and 2. It is common to order the labels in a priority queue after a non decreasing resource (see [35]). This will yield an algorithm where at most two labels $L_1, L_2 \in \mathcal{L}_v$ have $d(L_1) = d(L_2)$ and $t(L_1) = t(L_2)$.

In the special case where only one of the resources is used, that is $C = d_i = 0\ \forall i \in V_c$ or $a_i = b_i = 0 = T\ \forall i \in V_c$ and $\tau_{ij} = 0 \forall (i,j) \in A$. When only a single resource is present it is possible to reduce the size of $\mathcal{L}_v$ to 2. Assume that the capacity is the single resource used. Since labels are

processed in an increasing order it follows that the labels created in line 7-11 (Extended to node $v$) in algorithm 1 have capacity greater than or equal to any label in $\mathcal{L}_v$. Therefore dominance can be implemented by maintaining two labels with different predecessors in each set $\mathcal{L}_v, v \in V_c$.

Irnich and Villeneuve [35] have generalized the idea of cycle elimination and used an algorithm that ensured that the columns generated did not contain cycles of size $k$. Using an algorithm that eliminated cycles of size 3 they solved several unsolved VRPTW Solomon instances. In parallel with the success of Irnich and Villeneuve [35] the first successful BCP algorithm using elementary routes was proposed by Chabrier [12] therefore general k-cycle algorithms has not been studied in great details.

There exist several algorithms for solving the ESPPRC. If there are no cycles with negative cost in the graph $G$, then the ESPPRC is solvable in pseudo-polynomial time since connectivity of the path is implicitly ensured. In this particular case several algorithms based on dynamic programming exist for the ESPPCC, see e.g., Beasley and Christofides [7],Carlyle et al. [11], Dumitrescu and Boland [26], and Muhandiramge and Boland [50]. These algorithms exploit that when the capacity constraint is relaxed the corresponding problem is a regular shortest path problem with negative arcs. Since shortest path problem is polynomial solvable and is a valid lower bound it can be used to eliminate labels.

In the case where the graph may contain negative cycles, which is the case in the column generation context some of the most important events for the dynamic programming algorithms are as follows. Feillet et al. [27] present a dynamic programming algorithm where the elementary property of the path is ensured by use of an additional resource per node. Righini and Salani [57] proposed a general bi-directional approach to solve the ESPPRC. In a bi-directional algorithm a set of forward and backwards labels are computed and the labels from the two sets is then spliced together. In the case where the resources are capacity and time and the splice point is selected as $\frac{C}{2}$ in advance, it is possible to construct a graph (see [3] for details) for calculating the backward label set. With some adjustments of the generic framework (algorithm 1) the forward and backward label sets can be generated using this. Irnich and Desaulniers [34] characterized a resource using the so-called resource extension functions. As the name suggest the function defines how an extension by a label $L$ to a node $v$ is done. Irnich [33] analyses the properties of these functions. If all resource extension functions can be inverted it can be shown that there exist a bi-directional algorithm. Independently, Boland et al. [10] and Righini and Salani [58] proposed to initially relax the node resources and add them iteratively until the path is elementary. In the former paper this is referred to as a *state space augmentation* algorithm and in the latter it is denoted a *decremental state space relaxation* algorithm.

The standard implementation of a labeling algorithm for ESPPRC uses

a bit vector to store the node resources. A bit is set if a node has been visited or is unreachable. A node is unreachable for a label, if there does not exist a feasible extension of the label to that node in the graph. Let the function U(L) return the set of vertexes that have been either visited or are unreachable. The dominance criteria introduced by Feillet et al. [27] can then be stated as:

**Proposition 1.** *A label $L_1$ dominates a label $L_2$ if:*

$$c(L_1) \leq c(L_2) \quad d(L_1) \leq d(L_2) \quad t(L_1) \leq t(L_2) \quad U(L_1) \subseteq U(L_2)$$

The intuition behind the dominance in proposition 1 is, that any feasible extension for $L_2$ is feasible for $L_1$. Since $L_1$ has a lower cost than $L_2$ any feasible path constructed from the same extension will always have a lower cost when combined with $L_1$.

To reduce the number of labels Chabrier [12] suggested to adjust the cost of a label $L_1$ where $L_1 \nsubseteq L_2$. The cost adjustment was made by subtracting the dual variable $\pi_i \geq 0$ if the node was in the set $U(L_1)$ but not in $U(L_2)$. Through computational studies Chabrier [12] choose to restrict the number of bits the labels differed with to two bits.

Righini and Salani [57] proposed a bounding procedure to fathom labels. The idea is formalized in the following proposition:

**Proposition 2.** *Given a label $L$ and an upper bound $UB$ on the solution. If $L_{LB}$ is a lower bound on the value of any feasible extension of $L$ and $c(L) + L_{LB} \geq UB$. No extension of $L$ can lead to an optimal solution.*

The bounding procedure of Righini and Salani [57] calculates a lower bound for a label based on a linear knapsack relaxation for a single label. In the context of ESPPCC Baldacci et al. [1] suggest to use the 2SPPRC relaxation as bounding function. It should be noted that they precomputed the bounds in each node, by solving a single 2SPPRC . For the VRPTW Baldacci et al. [3] has proposed to use the so called NG-routes as a bounding, which yields better bounds and can also be precomputed.

### Cuts

It is possible to improve the quality of the lower bound obtained when solving the LP relaxation of the Set Partition. This is done by adding cutting planes in a approach similar to the one in BAC. In the context of VRP Kohl et al. [39] were among the first to improve the Set Partition formulation with the introduction of the K-Path Cuts for VRPTW. Desaulniers et al. [22] later generalized these inequalities and reported some good computational results for the Solomon instances. In the context of CVRP Fukasawa et al. [32] integrated the valid inequalities used by Lysgaard et al. [48] to solve the CVRP trough BAC. Common for the above valid inequalities is that they are

formulated in the two index formulation(1.1.1). When a violated inequality is found it is decomposed on the fly and kept in the master problem. The only consequence is that the objective of the pricing problem is modified. We refer to cuts formed in this fashion as cuts formed in the original solution space. Due to the collapse of the identical pricing problems into one, cuts formed in the original space must be identical for each vehicle, otherwise the property of identical pricing problems is lost. For a general discussion of cutting planes in BCP algorithms refer to Desaulniers et al. [23].

To cut in the original solution space the current solution of the master problem is transformed to a solution in the original space. In the case of VRP this poses some challenges since the pricing problems have been collapsed into a single pricing problem. Therefore a transformation is made back to a relaxed version of the two index edge/arc flow model from section 1.1.1. In this variant the variables $x_{ij}, (i,j) \in A$ are equal to the amount of flow on arc $(i,j) \in A$. Regardless of the pricing problem any solution to the master problem can then be transformed back to this space as follows:

$$x_{ij} = \sum_{p \in \mathcal{P}} \alpha_{ijp} \lambda_p \tag{1.36}$$

Once the master solution has been transformed to the original space the usual separation routines can be used.

When a violated inequality is found it needs to be decomposed into the master problem. As an example we shall consider how to handle the capacity constraints from section 1.1.1. For some $S \subseteq V_c, |S| \geq 2$, the undirected version of (1.12) is:

$$\sum_{(i,j) \in \delta(S)} x_{ij} \geq 2r(S)$$

When decomposed into the master problem the inequality becomes:

$$\sum_{p \in P} \sum_{(i,j) \in \delta(S)} \alpha_{ijp} \lambda_p \geq 2r(S)$$

To handle the cut in the pricing problem, let $\sigma \geq 0$ denote the dual of the constraint. Then the edge weights for any $(i,j) \in \delta(S)$ become:

$$w(i,j) = c_{ij} - \frac{1}{2}\pi_i - \frac{1}{2}\pi_j - \sigma, \quad (i,j) \in A$$

The pricing problem is then solved using these arc weights. New columns that use any arc $(i,j) \in \delta(S)$ are then added to the cut in the master problem when generated.

It is possible to formulate inequalities directly based on the variables in the Set Partition formulation. This was proposed independently by Baldacci et al. [1] who proposed the strong capacity inequalities and Jepsen et al.(chapter 2) who proposed the subset row inequalities (SR). The strong capacity inequalities can be defined as follows.

**Proposition 3.** *Given $S \subseteq V_c$ the inequality:*

$$\sum_{p \in \mathcal{P}: \sum_{(i,j)\in\delta(S)} \alpha_{ijp} \geq 1} \lambda_p \geq r(S)$$

*is valid*

The idea behind the strong capacity inequalities is the observation that even though a route may visit the set $S$ more than once, a route corresponds to a single vehicle visit and therefore it is valid only to account for it once. To handle the cuts in the pricing problem the dominance rule of the ESPPRC is modified as follows:

**Proposition 4.** *A label $L_1$ dominates a label $L_2$ if:*

$$c(L_1) \leq c(L_2) \quad d(L_1) \leq d(L_2) \quad t(L_1) \leq t(L_2) \quad U(L_1) = U(L_2)$$

The only difference between proposition 4 and 1 is that we now use an equality when comparing the set of unreachable nodes. This is needed since the cost of a feasible extension for $L_2$ may have a different cost for $L_1$. This difference in cost appears when $L_2$ has entered and left a set of a strong capacity cut but $L_1$ has not done that.

Jepsen et al.(Chapter 2) introduce the subset-row(SR) inequalities which can be stated as:

**Proposition 5.** *Given a $S \subseteq V_c$ and constant $0 < k \leq |S|$ the inequalities :*

$$\sum_{p \in P} \left\lfloor \frac{1}{k} \sum_{(i,j)\in\delta^+(S)} \alpha_{ijp} \right\rfloor \lambda_p \leq \left\lfloor \frac{|S|}{k} \right\rfloor$$

*are valid for the set partition formulation*

The idea behind the SR-inequalities is to count the number of routes that intersects the set $S$ a certain number of times. Whenever this number of routes becomes larger than the constant $\frac{|S|}{k}$ the set $S$ the number of routes that visit the set $S$ is to high. The dominance rule in proposition 5 can be used, but we will present an alternative dominance rule in Chapter 2. It is worth mentioning that the separation routines in chapter 2 only include inequalities where $|S| \leq 7$ and $k \leq 3$. Petersen et al. [53] has conducted computational results for Chvátal-Gomory Rank-1 Cuts and showed that the BCP algorithms were often able to solve the VRPTW instances in the root node. Desaulniers et al. [22] and Baldacci et al. [3] restrict the use of SR-inequalities to $|S| = 3$ and $k = 2$ which seems to yield BCP algorithms that perform better in practice. Spoorendonk and Desaulniers [61] studied how to handle clique inequalities but the computational results were not as good as the results obtained using the SR-inequalities.

It is important to note that when a bounding function is used, inequalities that have a positive dual variable associated must be taken into account in the relaxation. As an example, if a strong capacity constraint is added to the master problem with dual variable $\sigma \geq 0$, the dual is subtracted from all arcs in $\delta(S)$, when the relaxation is solved. An alternative solution is clearly to handle the cut directly in the relaxation but that may often be as difficult as handling it in the ESPPRC.

### Obtaining Integer Solutions

First, it is important to note that an feasible integer solution to the original VRP can be a fractional solution in the Set Partition model. Therefore a fractional solution should always be converted to the original model using the transformation in equation (1.36) and if the transformed solution is feasible the optimization is complete. Once it is verified that the current master solution is not feasible in the original space and that there is no more columns with negative reduced cost, an integer solution can be found using one of the following two methods:

- Traditional branch and bound where branching is done on an arc or a hyperplane in the original space.
- Using enumeration.

Similar to cutting branching in the original space, branching needs to be enforced in such a way that the property of the identical sub problems is not lost, for a general discussion on how to branch in a BCP algorithm with identical pricing problems we refer the reader to Vanderbeck [63]. The first branching rule in the BCP algorithms for the VRP is in principle hyperplane branches where it is enforced that either zero or one of the vehicles must use a given arc. This branch can be enforced by adding a cut in the master problem, but in early algorithms such as the one by Desrosiers et al. [24] the branch was incorporated in the pricing problem. Fukasawa et al. [32] extended the branch rule to use the traditional hyperplane branch know from the CVRP.

The other method is enumeration which has proven to be very successful for both CVRP[1] and VRPTW[3]. In enumeration an upper bound $UB$ and a lower bound $LB$ are used. From reduced cost fixing of a binary variable it is know that any non basic column with a reduced cost strictly greater than the gap $ub - lb$ can not be part of an integer solution which is an improvement of the current solution. This complete set of columns can be found by solving an ESPPRC using the dominance rule in proposition 5 and bounding functions. Once we have added the columns with reduced cost less than or equal to the gap the resulting problem can be solved as an integer optimization problem.

### 1.1.4 Exact versus Heuristic Solutions

Although the main focus of this thesis is exact solution methods, it is important to remember that in real life the computational time needed to find the optimal solution is not always available therefore it is important that the exact solution algorithms can find good solutions within reasonable time. Danna and Le Pape [18] have shown how to integrate the Branch-and-Price algorithm with a local search framework. This integration helps the BCP algorithm with finding good integer solutions in the early stages of the algorithm. The method has show to result in reasonable good solutions for the VRPTW. Prescott-Gagnon et al. [55] have improved the heuristic approach for BCP algorithms further by integrating the BCP algorithm with large neighbourhood search. For BAC algorithms methods such as local branching introduced by Fischetti and Lodi [28] and the feasibility pump introduced by Fischetti et al. [29] can be used to find fast and good solutions. The main benefit of the exact solution approach is that it provides both an upper and lower bound.

Though when a fast good solution is needed a heuristics such as the adaptive large scale neighbourhood search by Pisinger and Ropke [54] for CVRP, VRPTW and many other Vehicle Routing variants or the local search heuristic by Zachariadis and Kiranoudis [65] are preferable.

### 1.1.5 Applications of the BAC and BCP Algorithms

The solution methods for the Traveling Salesman Problem has served as a big inspiration in the first steps of the modern BAC and BCP algorithms for the CVRP and VRPTW. Today the BAC and BCP algorithms for these problems serve as sources of inspiration when new problems within the field of routing are encountered and are often a fundamental for the successful solution of these problems. Therefore an improvement of the BAC and BCP algorithms for CVRP and VRPTW does not only serve these problems, but may benefit a whole range of problems. Some examples of problems where the BAC and BCP algorithms have been a source of inspiration are the Split Delivery Vehicle Routing Problem, the Pickup-and-Delivery Vehicle Routing Problem and the Capacitated Location Routing Problem in the following section.

In the Split Delivery Vehicle Routing Problem the customers may be serviced by more than one vehicle. The problem is solved with a capacity bound as the CVRP and time windows are also included. Belenguer et al. [9] have developed lower bounds and a BAC algorithm where the model and cuts used in the BAC algorithm for CVRP serve as a major inspiration. Desaulniers [21] has developed a BCP algorithm where the solution approach for the ESPPRC has been a source of inspiration for the solution of the pricing problem. Furthermore, the $k$-path inequalities introduced for the VRPTW are

used as cutting planes.

In the Pickup-and-Delivery Vehicle Routing Problem a given quantity must be transported between a pickup point and a delivery point. The problem is typically studied with time windows and in this case Ropke et al. [59] have constructed a BAC algorithm where many of the aspects of the BAC algorithm for CVRP are incorporated. Ropke et al. [59] developed a BCP algorithm where the solution method of the pricing problem was inspired by the solution method for the ESPPRC and Baldacci et al. [4] extended the BCP algorithm for CVRP by Baldacci et al. [1].

## 1.2 Capacitated Location Routing Problem

The Capacitated Location Routing Problem(CLRP) is closely related to the two echelon vehicle routing problem considered in Chapter 3 and some of the ideas from CLRP have therefore been adapted to this problem. The Location Routing Problem (LRP) is a strategic planning problem where a set of distribution locations have to be selected and a set of routes connected to these location must be found. The goal is to minimize the sum of the cost for opening the locations, using the vehicles and traversing the arcs. The problem differs from the classic Facility Location problem [64] where the cost of connecting a customer to the facility is measured in the Euclidean distance. The first exact algorithm for LRP was a BAC algorithm introduced by Laporte and Nobert [41]. This algorithm was later extended to the more general case of the capacitated location routing problem by Laporte et al. [43]. Belenguer et al. [8] introduced several new cutting planes and improved the model and Contardo et al. [15] introduced a symmetric model and introduced new cutting planes. For a literature review on other variants of the problem and heuristics we refer the reader to Nagy and Salhi [52]. To formulate the problem mathematically let:

- $V_c$ define the set of customers
- $V_s$ define the set of locations
- $E$ define the set of edges.
- $c_e$ is the cost of using edge $e \in E$
- $x_e \in \{0, 1\},\ e \in E$ is 1 if the edge is used.
- $z_e \in \{0, 1\},\ e \in E$ is 1 if the edge between a satellite and a customer is used twice, the customer is visited on a single customer route.

For each location $r \in V_s$ let:

- $g_r$ be the fixed charge of using the location
- $f_r$ be the fixed charge of using a vehicle from a location

- $\hat{m}_r$ and $\underline{m}_r$ are respectively a lower and a upper bound on the number of vehicles that must/can be used at the location
- $y_r \in \{0, 1\}$ is 1 if the location is used and 0 otherwise
- $m_r$ is equal to the number of vehicles used at the facility

Finally let $\mathcal{P}$ denote the set of all paths between two locations $s_j, s_k \in V_s, s_j \neq s_k$. For a path $P \in \mathcal{P}$ let $e_1(P)$ be the first edge on the path and $e_l(P)$ the last edge on the path that is the edges from the two satellites to the customers on the path. The set of edges excluding the first and the last is defined as $\bar{P}$ that is the edges that connects the customers.

The mathematical model introduced by Laporte et al. [43] can then be formulated as:

$$\min \sum_{e \in E(V_c)} c_e x_e + \sum_{E(V_s:V_c)} z_e + \sum_{r \in V_s} (g_r y_r + f_r m_r)$$

subject to

$$\sum_{e \in \delta(i)} x_e + 2z_e = 2 \qquad \forall i \in V_c \qquad (1.37)$$

$$\sum_{e \in E(r:V_c)} x_e + 2z_e = 2m_r \qquad \forall r \in V_s \qquad (1.38)$$

$$\sum_{e \in \delta(S)} x_e \geq 2r(S) \qquad \forall S \subseteq V_c, |S| \geq 3 \qquad (1.39)$$

$$\sum_{E(V_s:i)} x_e \leq 1 \qquad \forall i \in V_c \qquad (1.40)$$

$$x_{e_1} + 3x_e + x_{e_2} \leq 4 \qquad \forall e_1, e_2 \in E(V_s : V_c), e_1 \neq e_2, \forall e \in E(V_c) \qquad (1.41)$$

$$x_{e_1(P)} + x_{e_l(P)} + 2\sum_{e \in \bar{P}} x_e \leq 2|P| - 5 \qquad \forall P \in \mathcal{P}, |P| \geq 3 \qquad (1.42)$$

$$y_r \leq m_r \leq |V_c| y_r \qquad \forall r \in V_s \qquad (1.43)$$

$$\hat{m}_r \leq m_r \leq \underline{m}_r \qquad \forall r \in V_s \qquad (1.44)$$

$$x_e \in \{0, 1\} \qquad \forall e \in E(V_c) \qquad (1.45)$$

$$z_e \in \{0, 1\} \qquad \forall e \in E(V_s : V_c) \qquad (1.46)$$

$$y_r \in \{0, 1\} \qquad \forall r \in V_s \qquad (1.47)$$

The objective minimizes the sum of the routing cost, location cost and start up cost of vehicles. Constraints (1.37) ensure that each customer is visited, constraints (1.38) binds the number of used vehicles to the number of entering and leaving edges of the location. Constraints (1.39) eliminate sub tours and ensure that a set is visited by the number of vehicles needed. Constraints (1.40) to constraints (1.42) ensures that a route starts and ends

in the same location. Constraints (1.43) force at least one vehicle out of a location that is used and ensures that the location is used when a vehicle leaves it. Constraints (1.44) impose bounds on the number of vehicles that can be used from a location. Finally constraints (1.45) to (1.47) are the domains of the variables. Constraints

(1.42) are commonly referred to as the *chain barring* constraints. Figure 1.1 (a) shows a solution which is feasible to the model for CLRP if constraints (1.42) are not included. The solution is then cut of by adding a *chain barring* constraint for the path $P = \{1, 2, 3, 4, 5\}$. Although the addition of the *chain barring* leads to an optimal integer solution the inequalities are weak when the flow on the edges are not integer. Such an issue is illustrated on figure 1.1 (b) where there are no violated *chain barring* constraints. As the reader may note it is not feasible for a customer to use edges to two different locations. This observation has led to the development of the *path elimination constraint* which was introduced by Belenguer et al. [8]:

$$\sum_{e \in \delta(S)} x_e \geq \sum_{e \in E(\{i\}:I)} 2x_e + \sum_{e \in E(\{j\}:V_s \setminus I)} 2x_e \forall S \subseteq V_c, |S| \geq 2, \forall i, j \in S, \forall I \subset V_s \tag{1.48}$$

The idea of the constraint is to partition the locations into two sets and then enforce that if both customers are connected to a location (In a fractional solution it may be more than one) then the flow out of the set is at least twice the size of the flow from the two location sets. For a formal proof of the constraints the reader is referred to Belenguer et al. [8]. By selecting the sets $I = \{1, 6\}$ and $S = \{2, 3, 4\}$ and the customers as $i = 2$ and $j = 4$ a violated *path elimination constraint* constraint is obtained and the solution can be cut off.

Belenguer et al. [8] has shown that cutting planes valid for the CVRP can be used for the CLRP. To separate valid inequalities for the CVRP all locations are contracted into a single depot. When the cut is added the contracted edges are then expanded to obtain a valid inequality. A similar approach is used for the 2ECVRP in section 3.

Additional valid inequalities has been introduced for the CLRP by both Belenguer et al. [8] and Contardo et al. [15]. Some of these may be valid for the 2ECVRP and a future research direction for this problem could therefore be to adapt these inequalities.

## 1.3 Overview of Thesis

The remaining part of the thesis chapters contains a set of papers which have been constructed in collaboration with many authors and a conclusion in chapter 7. Each chapter has been written as a self containing paper, but especially Chapter 6 is dependant on the techniques in some of the other chapters. Therefore this Chapter has been placed as the final Chapter of

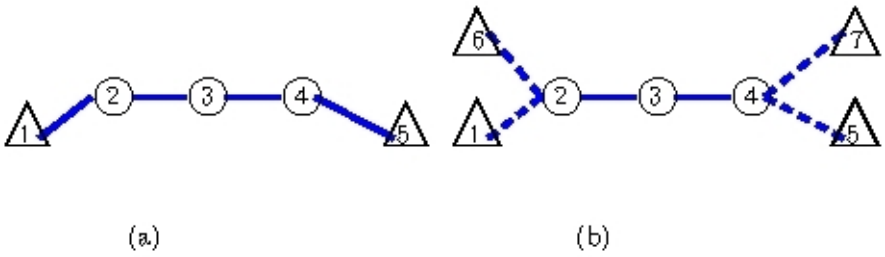



Figure 1.1: Illustration of customer connections to locations. Triangles are locations and circles are customers. Solid lines indicate $x_e = 1$ and dashed lines indicate $x_e = 0.5$. (a) shows an example of a solution that violate constraints (1.42) and (b) shows an example of a solution that violate constraints (1.48), but not any of constraints (1.42)

the thesis and the two Chapters which it relates to proceeds it. The final two chapters are ordered Chronological after finish date. In the following a summary of each of the chapters is given:

**Chapter 2: Subset-Row Inequalities Applied to the Vehicle Routing Problem with Time Windows** The paper considers the classical Danzig-Wolfe decomposition of VRPTW and develops a BCP algorithm, where cuts are added to the master problem. The cuts in the master problem are the SR-inequalities, which are valid inequalities for the set packing polytope. The cuts are derived as a Chvátal Gomory cut, but possesses some good properties that make it possible to integrate them into the dominance criteria of the ESPPRC. The new algorithm for VRPTW was able to solve 8 previously unsolved instances and. The paper is published in *Operation Research*.

**Chapter 3: A Branch-and-Cut Algorithm for the Symmetric Two-echelon Capacitated Vehicle Routing Problem** . The 2ECVRP considers distribution of goods from the depot to the customers through a set of satellites. Two sets of vehicle types are considered, a large capacity vehicle type going between the depot and the satellites and a small capacity vehicle type servicing the customers from the satellites. If the two echelon is considered interdependently, the first echelon is a split delivery problem and the second echelon is a Location Routing Problem. The paper introduces a model which is a lower bound for the 2ECVRP, adapts cutting planes from the CLRP and CVRP and devises a specialized branching rule to obtain an integer feasible solution. The paper is accepted in *Transportation Science* with revisions.

**Chapter 4: A Branch-and-Cut Algorithm for the Capacitated Profitable Tour Problem** The Capacitated Profitable Tour Problem(CPTP) is closely related to the ESPPRC. The difference is that a simple cycle is to be found, such that the sum of the capacities of the nodes on the cycle does

not exceed a given threshold and the cost of the used edges minus the profit of the nodes is minimized. The paper introduces many cutting planes from related polytopes, as well as several new cutting planes based on the GLM and KLM inequalities described in section 1.1.1. The computational results show that instances with up to 800 nodes can be solved to optimality within an hour of CPU time. The paper is submitted.

**Chapters 5: Partial Path Column Generation for the Vehicle Routing Problem** The paper introduces a new formulation for the CVRP and VRPTW, where small path segments are combined to form a solution. Using a Dantzig-Wolfe reformulation a master problem which contains additional constraints(Compared to the Set Partition Model) and a ESPPRC pricing problem is formed. In the pricing problem one of the resources is reduced which reduces the state-space used when solving the problem with dynamic programming. The BCP algorithm is tested and computational results are reported. The chapter is based on a conference papers presented at INOC09, which can be found in appendix C.

**Chapter 6: Partial Path Column Generation for the Elementary Shortest Path Problem with a Capacity Constraints** The paper considers an alternative formulation for the ESPPRC. The formulation is related to the ideas presented in Chapter 5 but the master and pricing problem are slightly different. Both the master and pricing problems have a structure similar to the ESPPRC. Computational results are presented and discussed . The chapter is based on a conference papers presented at INOC09 and has been extended with some computational results. Furthermore references to the papers in this thesis has been changed to refer to the relevant chapter rather than the preliminary technical reports.

**Appendix A: The vehicle routing problem with edge set costs** The paper introduces an extension of the VRPTW where there is a penalty associated with different subset of edges. The problem is solved using the Branch-and-Cut-and-Price algorithm developed in Chapter 2 as a key component. The key in the solution approach is that the constraints associated with the set penalties are kept in the master problem, which leads to a solution method where the standard pricing problem of VRPTW is solved. A set of benchmark instances based on the Solomon instances are introduced and computational results are presented. The paper is submitted.

**Appendix B: A Path Based Model for a Green Liner Shipping Network Design Problem** The paper contains a new formulation for the Liner Shipping Network Design Problem, where the network rotation

patterns are generated using a dynamic programming algorithm. The algorithmic approaches are similar to the solution approach used in the BCP algorithms for the CVRP and VRPTW and the dynamic programming algorithm for generating the rotations is also inspired by the solution methods for the ESPPRC. The paper is an extended version of the conference paper for the 2011IANENG International Conference on Operational Research.

**General Overview** The chapters in Part I and II gives an overview of how the two solution methods BAC and BCP can be used in the context of Vehicle Routing and how they can be improved. The thesis considers six different problems. In many of the papers the standard methods known from the literature are either changed or improved significantly. The chapters gives an overview on how to apply and improve the BCP and BAC algorithms known from the CVRP and VRPTW.

# Bibliography


[1] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 10 2008.

[2] R. Baldacci, E. Bartolini, A. Mingozzi, and R. Roberti. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science*, 7(3):229–268, 2010.

[3] R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. Working paper, 2010.

[4] R. Baldacci, E. Bartolini, and A. Mingozzi. An exact algorithm for the pickup and delivery problem with time windows. *Operations Research*, 59(2):414–426, 2011. cited By (since 1996) 0.

[5] M. L. Balinski and R. E. Quandt. On an integer program for a delivery problem. *OPERATIONS RESEARCH*, 12(2):300–304, 1964. doi: 10.1287/opre.12.2.300. URL `http://or.journal.informs.org/cgi/content/abstract/12/2/300`.

[6] J. F. Bard, G. Kontoravdis, and G. Yu. A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36(2):250–269, May 2002.

[7] J.E. Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19:379–394, 1989. doi: 10.1002/net.3230190402.

[8] J.-M. Belenguer, E. Benavent, C. Prins, C. Prodhon, and R.W. Calvo. A branch-and-cut method for the capacitated location-routing problem. *Computers & Operations Research*, 38:931–941, 2011.

[9] J.M. Belenguer, M.C. Martinez, and E. Mota. A lower bound for the split delivery vehicle routing problem. *Operations research*, 48(5):801–810, 2000.

[10] N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operation Research Letters*, 34(1):58–68, 2006. doi: 10.1016/j.orl.2004.11.011.

[11] W.M. Carlyle, J.O. Royset, and R.K. Wood. Lagrangian relaxation and enumeration for solving constrained shortest-path problems. *Networks*, 51(3):155–170, 2008. doi: 10.1002/net.20212.

[12] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 33(10): 2972–2990, 2006. doi: 10.1016/j.cor.2005.02.029.

[13] N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for the vehicle rout- ing problem based on spanning tree and shortest path relaxation. *Mathematical Programming*, 10:255–280, 1981.

[14] N. Christofides, A. Mingozzi, and P. Toth. State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11(2):145–164, 1981. ISSN 1097-0037. doi: 10.1002/net.3230110207. URL `http://dx.doi.org/10.1002/net.3230110207`.

[15] C. Contardo, J.-F. Cordeau, and B. Gendron. A branch-and-cut algorithm for the capacitated location-routing problem. Submitted for publication in CAOR, 10 2010.

[16] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001. ISBN 0070131511.

[17] G Cornuejols and F. Harche. Polyhedral study of the capacitated vehicle routing problem. *Mathematical Programming*, 60:21–52, 1993. ISSN 0025-5610. URL `http://dx.doi.org/10.1007/BF01580599`. 10.1007/BF01580599.

[18] E. Danna and C. Le Pape. Branch-and-price heuristics: A case study on the vehicle routing problem with time windows. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, chapter 4, pages 99–129. Springer, 2005. doi: 10.1007/0-387-25486-2_4.

[19] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *MANAGEMENT SCIENCE*, 6(1):80–91, /10/1 1959. URL `http://mansci.journal.informs.org/cgi/content/abstract/6/1/80`. 10.1287/mnsc.6.1.80.

[20] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

[21] G. Desaulniers. Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations research*, 58(1):179–192, 2010.

[22] G. Desaulniers, F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. *TRANSPORTATION SCIENCE*, 42(3):387–404, 2008. doi: 10.1287/trsc.1070.0223. URL `http://transci.journal.informs.org/cgi/content/abstract/42/3/387`.

[23] G. Desaulniers, J. Desrosiers, and S. Spoorendonk. Cutting planes for branch-and-price algorithms. *Networks*, 2011. accepted.

[24] J. Desrosiers, F. Soumis, and M. Desrochers. Routing with time windows by column generation. *Networks*, 14:545–565, 1984.

[25] M. Dror. Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, 42:977–979, 1994.

[26] I. Dumitrescu and N. Boland. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks*, 42(3):135–153, 2003. doi: 10.1002/net.10090.

[27] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004. doi: 10.1002/net.v44:3.

[28] M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98:23–47, 2003. ISSN 0025-5610. URL `http://dx.doi.org/10.1007/s10107-003-0395-5`. 10.1007/s10107-003-0395-5.

[29] M. Fischetti, F. Glover, and A. Lodi. The feasibility pump. *Mathematical Programming*, 104(1):91–104, 2005. cited By (since 1996) 29.

[30] L. Marshall Fisher and Ramchandran Jaikumar. Generalized assignment heuristic for vehicle routing. *Networks*, 11(2):109–124, 1981. cited By (since 1996) 188.

[31] B A Foster and D M Ryan. An integer programming approach to the vehicle scheduling problem. *Operational Research Quarterly*, 27(2): 367–384, 1976. URL `http://www.jstor.org/stable/3009018`.

[32] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R.F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming, Ser. A*, 106(3):491–511, 2006.

[33] S. Irnich. Resource extension functions: properties, inversion, and generalization to segments. *OR Spectrum*, 30:113–148, 2008. ISSN 0171-6468. URL http://dx.doi.org/10.1007/s00291-007-0083-6. 10.1007/s00291-007-0083-6.

[34] S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, Jacques Desrosiers, and M.M. Solomon, editors, *Column Generation*, chapter 2, pages 33–65. Springer, 2005. doi: 10.1007/0-387-25486-2_2.

[35] S. Irnich and D. Villeneuve. The shortest-path problem with resource constraints and k-cycle elimination for k $\geq$ 3. *INFORMS J. on Computing*, 18(3):391–406, 2006.

[36] B. Kallehauge, J. Larsen, and O.B.G. Madsen. Lagrangian duality applied to the vehicle routing problem with time windows. *Computers and Operations Research*, 33(5):1464–1487, 2006. cited By (since 1996) 23.

[37] B. Kallehauge, N. Boland, and Madsen O.B.G. Path inequalities for the vehicle routing problem with time windows. *Networks*, 49(4):273–293, 2007. cited By (since 1996) 4.

[38] N. Kohl and O.B.G. Madsen. An optimization algorithm for the vehicle routing problem with time windows based on lagrangian relaxation. *Operations Research*, 45(3):395–406, 1997. cited By (since 1996) 64.

[39] N. Kohl, J. Desrosiers, O. B. G. Madsen, Marius M. Solomon, and Francois Soumis. 2-path cuts for the vehicle routing problem with time windows. *TRANSPORTATION SCIENCE*, 33(1):101–116, 1999. doi: 10.1287/trsc.33.1.101. URL http://transci.journal.informs.org/cgi/content/abstract/33/1/101.

[40] A. W. J. Kolen, A. H. G. Rinnooy Kan, and H. W. J. M. Trienekens. Vehicle routing with time windows. *Operations Research*, 35:266–273, 1987.

[41] G. Laporte and Y. Nobert. An exact algorithm for minimizing routing and operating costs in depot location. *European Journal of Operational Research*, 6(2):224–226, 2 1981.

[42] G. Laporte and Y. Nobert. A branch and bound algorithm for the capacitated vehicle routing problem. *OR Spektrum*, 5:77–85, 1983.

[43] G. Laporte, Y. Nobert, and D. Arpin. An exact algorithm for solving a capacitated location-routing problem. *Annals of Operations Research*, 6(9):291–310, 1986. Cited By (since 1996): 21.

[44] A. N. Letchford, R. W. Eglese, and J. Lysgaard. Multistars, partial multistars and the capacitated vehicle routing problem. *Mathematical Programming, Series B*, 94(1):21–40, 2002.

[45] A.N. Letchford and J.-J. Salazar-González. Projection results for vehicle routing. *Mathematical Programming*, 105(2-3):251–274, 2006.

[46] N. Letchford, A and J. Salazar-González. Projection results for vehicle routing. *Mathematical Programming*, 105:251–274, 2006. ISSN 0025-5610. URL `http://dx.doi.org/10.1007/s10107-005-0652-x`. 10.1007/s10107-005-0652-x.

[47] J. Lysgaard. The cvrpsep package. http://www.hha.dk/ lys/CVRPSEP.htm, 2003.

[48] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423–445, 2004.

[49] C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *J. ACM*, 7:326–329, October 1960. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/321043.321046. URL `http://doi.acm.org/10.1145/321043.321046`.

[50] R. Muhandiramge and N. Boland. Simultaneous solution of lagrangean dual problems interleaved with preprocessing for the weight constrained shortest path problem. *Networks*, 2009. doi: 10.1002/net20292.

[51] Denis Naddef and Giovanni Rinaldi. Branch-and-cut algorithms for the capacitated vrp. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*, chapter 3, pages 53–84. SIAM, Philadelphia, 2002.

[52] G. Nagy and S. Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177(2):649 – 672, 2007. ISSN 0377-2217. doi: DOI: 10.1016/j.ejor.2006.04.004.

[53] B. Petersen, D. Pisinger, and S. Spoorendonk. Chvátal-Gomory rank-1 cuts used in a Dantzig-Wolfe decomposition of the vehicle routing problem with time windows. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 397–420. Springer, 2008.

[54] D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):2403–2435, 2007.

[55] E. Prescott-Gagnon, G. Desaulniers, and L.-M. Rousseau. A branch-and-price-based large neighborhood search algorithm for the vehicle routing problem with time windows. *Networks*, 54(4):190–204, 2009. cited By (since 1996) 4.

[56] M. R. Rao and S. Zionts. Allocation of transportation units to alternative trips–a column generation scheme with out-of-kilter subproblems. *OPERATIONS RESEARCH*, 16(1):52–63, 1968. doi: 10.1287/opre.16.1.52. URL `http://or.journal.informs.org/cgi/content/abstract/16/1/52`.

[57] G. Righini and M. Salani. Symmetry helps: bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006. doi: 10.1016/j.disopt.2006.05.007.

[58] G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained shortest path problem. *Networks*, 51(3):155–170., 2008. doi: 10.1002/net.20212.

[59] S. Ropke, Cordeau J.-F., and G. Laporte. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, 49(4):258–272, 2007. cited By (since 1996) 23.

[60] M M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35:254–265, 1987.

[61] S. Spoorendonk and G. Desaulniers. Clique inequalities applied to the vehicle routing problem with time windows. *INFOR*, 48(1):53–67, 2010. cited By (since 1996) 0.

[62] P. Toth and D. Vigo, editors. *The vehicle routing problem*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001. ISBN 0-89871-498-2.

[63] F. Vanderbeck. Branching in branch-and-price: a generic scheme. *Mathematical Programming*, pages 1–46, 2010. ISSN 0025-5610. URL `http://dx.doi.org/10.1007/s10107-009-0334-1`. 10.1007/s10107-009-0334-1.

[64] L. A. Wolsey. *Integer Programming*. John Wiley & Sons, Inc., 1998.

[65] E. E. Zachariadis and C. T. Kiranoudis. A strategy for reducing the computational complexity of local search-based methods for the vehicle routing problem. *Computers & Operations Research*, 37(12):2089 – 2105, 2010. ISSN 0305-0548. doi: DOI: 10.1016/j.cor.2010.02.009.

# Part I

# Academic Papers

# Chapter 2 Subset-Row Inequalities Applied to the Vehicle Routing Problem with Time Windows

Mads Jepsen Bjørn Petersen Simon Spoorendonk and David Pisinger

*Department of Computer Science (DIKU), University of Copenhagen, DK-2100 Copenhagen Ø, Denmark*

April 2008

## Abstract

This paper presents a branch-and-cut-and-price algorithm for the vehicle routing problem with time windows. The standard Dantzig-Wolfe decomposition of the arc flow formulation leads to a set partitioning problem as the master problem and an elementary shortest path problem with resource constraints as the pricing problem. We introduce the subset-row inequalities, which are Chvatal-Gomory rank-1 cuts based on a subset of the constraints in the master problem. Applying a subset-row inequality in the master problem increases the complexity of the label-setting algorithm used to solve the pricing problem since an additional resource is added for each inequality. We propose a modified dominance criterion that makes it possible to dominate more labels by exploiting the step-like structure of the objective function of the pricing problem. Computational experiments have been performed on the Solomon benchmarks where we were able to close several instances. The results show that applying subset-row inequalities in the master problem significantly improves the lower bound, and in many cases makes it possible to prove optimality in the root node.

## 2.1 Introduction

The vehicle routing problem with time windows (VRPTW) can be described as follows: A set of customers, each with a demand, needs to be serviced by a number of vehicles all starting and ending at a central depot. Each customer must be visited exactly once within a given time window, and the capacity of the vehicles must not be exceeded. The objective is to service

all customers traveling the least possible distance. In this paper we consider a homogenous fleet, i.e., all vehicles are identical.

The standard Dantzig-Wolfe decomposition of the arc flow formulation of the VRPTW is to split the problem into a master problem (a set partitioning problem) and a pricing problem (an elementary shortest path problem with resource constraints (ESPPRC), where capacity and time are the constrained resources). A restricted master problem can be solved with delayed column generation and embedded in a branch-and-bound framework to ensure integrality. Applying cutting planes either in the master or the pricing problem leads to a branch-and-cut-and-price algorithm (BCP).

Kohl et al. [23] implemented a successful BCP algorithm for the VRPTW by applying subtour elimination constraints and *two-path* cuts. Cook and Rich [8] generalized the two-path cuts to the *k-path* cuts. Common for these BCP algorithms is that all applied cuts are valid inequalities for the VRPTW, i.e., the *original* arc flow formulation, and contain a structure making it possible to handle values of the dual variables in the pricing problem without increasing the complexity of the problem. Fukasawa et al. [17] refer to this as a *robust* approach in their paper, where a range of valid inequalities for the capacitated vehicle routing problem are used in a BCP algorithm. The topic of column generation and BCP algorithms has been surveyed by Barnhart et al. [3] and Lubbecke and Desrosiers [25].

Dror [13] showed that the ESPPRC is strongly $\mathcal{NP}$-hard, hence a relaxation of the ESPPRC was used as a pricing problem in earlier BCP approaches for the VRPTW. The relaxed pricing problem where non-elementary paths are allowed is denoted the shortest path problem with resource constraints (SPPRC) and can be solved in pseudo-polynomial time using a label-setting algorithm, which was initially done by Desrochers [12]. To improve lower bounds of the master problem, Desrochers et al. [10] used 2-cycle elimination, which was later extended by Irnich and Villeneuve [20] to $k$-cycle elimination ($k$-cyc-SPPRC) where cycles containing $k$ or less nodes are not permitted.

Beasley and Christofides [4] proposed to solve the ESPPRC using Lagrangian relaxation. However, recently label-setting algorithms have become the most popular approach to solve the ESPPRC; see e.g. Dumitrescu [14] and Feillet et al. [16]. When solving the ESPPRC with a label-setting algorithm a binary resource for each node is added, which increases the complexity of the algorithm compared to solving the SPPRC or the $k$-cyc-SPPRC. Righini and Salani [31] developed a label-setting algorithm using the idea of Dijkstra's bi-directional shortest path algorithm that expands both forward and backward from the depot and connects routes in the middle, thereby potentially reducing the running time of the algorithm. Furthermore Righini and Salani [31] and Boland et al. [5] proposed a decremental state space algorithm that iteratively solves a SPPRC by applying resources that force nodes to be visited at most once. Recently Chabrier [7], Danna and Pape

[9], and Salani [32] successfully solved several previously unsolved instances of the VRPTW from the benchmarks of Solomon [33] using a label-setting algorithm for the ESPPRC.

In this paper, we extend the BCP framework to include valid inequalities for the master problem, more specifically by applying the subset-row (SR) inequalities to the set partitioning master problem. Nemhauser and Park [28] developed a similar BCP algorithm for the edge coloring problem, but to our knowledge no such algorithms for the VRPTW have been presented. Applying the SR inequalities leads to an increased complexity of the pricing problem since each inequality is represented by an additional resource. To improve the performance of the label-setting algorithm, we introduce a modified dominance criterion that handles the reduced cost calculation in a reasonable way. Moreover, the SR inequalities potentially provide better lower bounds and smaller branch trees.

The paper is organized as follows: In Section 2.2 we give an overview of the Dantzig-Wolfe decomposition of the VRPTW and describe how to calculate the reduced cost of columns when column generation is used. In Section 2.3 we introduce the SR inequalities and show that the separation problem is $\mathcal{NP}$-complete. In Section 2.4 we review the basics of a label-setting algorithm for solving the ESPPRC and show how to handle the modified pricing problem in the same label-setting algorithm. For details regarding label-setting algorithms (including bi-directionality) we refer to Desaulniers et al. [11], Irnich and Desaulniers [19], Irnich [18], Righini and Salani [31]. An algorithmic outline and computational results, using the Solomon benchmark instances, are presented in Section 2.5. Section 2.6 concludes the paper.

## 2.2 Decomposition

Let $C$ be the set of customers, let the set of nodes be $V = C \cup \{o, o'\}$ where $\{o\}$ denotes the depot at the start of the routes and $\{o'\}$ denotes the depot at the end; and let $E = \{(i,j) : i,j \in V,\ i \neq j\}$ be the edges between the nodes. Let $K$ be the set of vehicles with $|K|$ unbounded, each vehicle having capacity $D$, and let $d_i$ be the demand of customer $i \in C$ and $d_o = d_{o'} = 0$. Let $a_i$ be the beginning and $b_i$ be the end of the time window for node $i \in V$. Let $s_i$ be the service time for $i \in V$ and let $t_{ik}$ be the time vehicle $k \in K$ visits node $i \in V$, if $k$ visits $i$. Let $c_{ij}$ be the travel cost on edge $(i,j) \in E$ and let $x_{ijk}$ be a variable indicating whether vehicle $k \in K$ traverses edge $(i,j) \in E$. Last let $\tau_{ij} = c_{ij} + s_i > 0$ be the travel time on edge $(i,j) \in E$ plus the service time of customer $i$. The three-index flow model (Toth and

Vigo [35]) for the VRPTW is:

$$\min \sum_{k \in K} \sum_{(i,j) \in E} c_{ij} x_{ijk} \tag{2.1}$$

$$\text{s.t.} \sum_{k \in K} \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 1 \qquad \forall i \in C \tag{2.2}$$

$$\sum_{(i,j) \in \delta^+(o)} x_{ijk} = \sum_{(i,j) \in \delta^-(o')} x_{ijk} = 1 \qquad \forall k \in K \tag{2.3}$$

$$\sum_{(j,i) \in \delta^-(i)} x_{jik} - \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 0 \qquad \forall i \in C,\ \forall k \in K \tag{2.4}$$

$$\sum_{(i,j) \in E} d_i x_{ijk} \leq D \qquad k \in K \tag{2.5}$$

$$a_i \leq t_{ik} \leq b_i \qquad \forall i \in V,\ \forall k \in K \tag{2.6}$$

$$x_{ijk}(t_{ik} + \tau_{ij}) \leq t_{jk} \qquad \forall (i,j) \in E,\ \forall k \in K \tag{2.7}$$

$$x_{ijk} \in \{0,1\} \qquad \forall (i,j) \in E,\ \forall k \in K \tag{2.8}$$

Here (2.2) ensures that every customer $i \in C$ is visited, while (2.3) ensures that each route starts and ends in the depot. Constraint (2.4) maintains flow conservation, while (2.5) ensures that the capacity of each vehicle is not exceeded. Constraints (2.6), (2.7) ensure that the time windows are satisfied. Note that (2.7) together with the assumption that $\tau_{ij} > 0$ for all $(i,j) \in E$ eliminates sub-tours. The last constraints define the domain of the arc flow variables. Note that a zero-cost edge $x_{oo'k}$ between the start and end depot must be present for all vehicles for (2.3) to hold if not all vehicles are used.

The standard Dantzig-Wolfe decomposition of the VRPTW, see e.g. Desrochers et al. [10], leads to the following master problem:

$$\min \sum_{p \in P} \sum_{(i,j) \in E} c_{ij} \alpha_{ijp} \lambda_p \tag{2.9}$$

$$\text{s.t} \sum_{p \in P} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp} \lambda_p = 1 \qquad \forall i \in C \tag{2.10}$$

$$\lambda_p \in \{0,1\} \qquad \forall p \in P \tag{2.11}$$

where $P$ is the set of all feasible routes, the binary constant $\alpha_{ijp}$ is one if and only if edge $(i,j)$ is used by route $p \in P$, and the binary variable $\lambda_p$ indicates whether route $p$ is used. The master problem can be recognized as a set partitioning problem, and the LP relaxation may be solved using delayed column generation. Let $\pi \in \mathbb{R}$ be the dual variables of (2.10) and let $\pi_0 = 0$. Then the reduced cost of a route $p$ is:

$$\bar{c}_p = \sum_{(i,j) \in E} c_{ij} \alpha_{ijp} - \sum_{(i,j) \in E} \pi_j \alpha_{ijp} = \sum_{(i,j) \in E} (c_{ij} - \pi_j) \alpha_{ijp} \tag{2.12}$$

The pricing problem becomes an ESPPRC where the cost of each edge is $\overline{c}_{ij} = c_{ij} - \pi_j$ for all edges $(i,j) \in E$. When applying cuts during column generation we will distinguish between valid inequalities for the VRPTW constraints (2.2)-(2.8) and valid inequalities for the set partitioning constraints (2.10)-(2.11).

Consider a valid inequality for the VRPTW constraints (2.2)–(2.8) in terms of the arc flow variables $x$:

$$\sum_{k \in K} \sum_{(i,j) \in E} \beta_{ij} x_{ijk} \leq \beta_0 \tag{2.13}$$

When decomposed into the master problem, inequality (2.13) is reformulated as:

$$\sum_{p \in P} \sum_{(i,j) \in E} \beta_{ij} \alpha_{ijp} \lambda_p \leq \beta_0 \tag{2.14}$$

Let $\mu \leq 0$ be the dual variable of (2.14). The reduced cost of a column $p$ is then

$$\begin{aligned} \overline{c}_p &= \sum_{(i,j) \in E} c_{ij} \alpha_{ijp} - \sum_{(i,j) \in E} \pi_j \alpha_{ijp} - \mu \sum_{(i,j) \in E} \beta_{ij} \alpha_{ijp} \\ &= \sum_{(i,j) \in E} (c_{ij} - \pi_j - \mu \beta_{ij}) \alpha_{ijp} \end{aligned} \tag{2.15}$$

Compared to (2.12) an additional coefficient $\mu\beta_{ij}$ is subtracted from the cost of edge $(i,j)$ and the complexity of the pricing problem remains unchanged if we use the edge costs $\overline{c}_{ij} = c_{ij} - \pi_j - \mu\beta_{ij}$.

Now, consider adding a valid inequality for the set partitioning master problem (2.10)–(2.11) that cannot be written as a linear combination of the arc flow variables:

$$\sum_{p \in P} \beta_p \lambda_p \leq \beta_0 \tag{2.16}$$

Let $\sigma \leq 0$ be the dual variable of (2.16). The reduced cost of a column $p$ is:

$$\hat{c}_p = \overline{c}_p - \sigma \beta_p = \sum_{(i,j) \in E} \overline{c}_{ij} \alpha_{ijp} - \sigma \beta_p \tag{2.17}$$

In addition to the reduced cost computed for a column $p$ in (2.15) the cost $-\sigma\beta_p$ must be considered. To reflect the possible extra cost $-\sigma\beta_p$ it may be necessary to modify the pricing problem by adding constraints or variables, thereby increasing its complexity.

## 2.3 Subset-Row Inequalities

The set of valid inequalities for the set packing problem is a subset of the set of valid inequalities for the set partitioning problem since the latter problem is a special case of first-mentioned. Two well-known valid inequalities for the set packing problem are the clique and the odd-hole inequalities, where the first is known to be facet-defining for the set partitioning problem (Nemhauser and Wolsey [29]).

Since the master problem is a set partitioning problem, it would be obvious to go in this direction when looking for valid inequalities for the master problem. Consider the separation of a clique or an odd-hole inequality. The undirected conflict graph $G'(P, E')$ is defined as follows: Each column is a vertex in $G'$ and the edge set is given as:

$$E' = \left\{ (p,q) : \sum_{(i,j)\in\delta^+(i)} \alpha_{ijp} = 1 \wedge \sum_{(i,j)\in\delta^+(i)} \alpha_{ijq} = 1,\ i \in C,\ p,q \in P,\ p \neq q \right\}$$

That is, an edge is present if the two columns $p$ and $q$ have coefficient one in the same row. In a VRPTW context it reads: Two routes are conflicting if they are visiting the same customer. A clique in $G'$ leads to the valid clique inequality:

$$\sum_{p\in\hat{P}} \lambda_p \leq 1 \tag{2.18}$$

where $\hat{P} \subseteq P$ are the columns corresponding to the vertices of a clique in $G'$. A cycle visiting an odd number of vertices $P$ in $G'$ leads to the valid odd-hole inequality:

$$\sum_{p\in\hat{P}} \lambda_p \leq \left\lfloor \frac{|\hat{P}|}{2} \right\rfloor \tag{2.19}$$

where $\hat{P} \subseteq P$ are the columns corresponding to the vertices visited on the cycle in $G'$. However, when column generation is applied, it is not obvious how to reflect the reduced cost of (2.18) or (2.19) in the pricing problem since there is no specific knowledge of the columns of the master problem when solving the pricing problem.

Inspired by the above inequalities (2.18) and (2.19) we introduce the *subset-row inequalities* (SR inequalities). These inequalities are specifically linked to the rows (rather than the columns) of the set packing problem, hence making it possible to identify the coefficient of a column in an SR inequality.

**Definition 1.** *Consider the set packing structure*

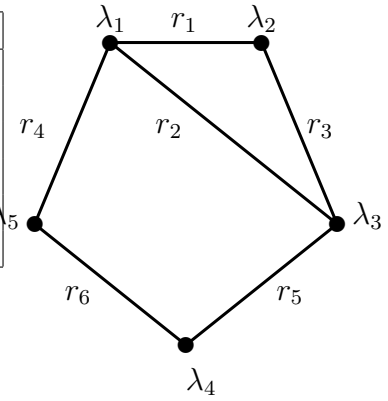
$$X = \{\lambda \in \mathbb{B}^{|P|} : A\lambda \leq 1\} \tag{2.20}$$

SR inequalities derived from the conflict graph of a set packing problem. In the LP-solution to $A\lambda \leq 1$ all $\lambda$ variables are $\frac{1}{2}$, which results in two violated SR inequalities:

- With $|S| = 3$ and $k = 2$ due to variables $\lambda_1$, $\lambda_2$, and $\lambda_3$ giving the set of rows $S = \{r_1, r_2, r_3\}$
- With $n = 5$ and $k = 2$ due to variables $\lambda_1$, $\lambda_2$, $\lambda_3$, $\lambda_4$, and $\lambda_5$ giving the set of rows $S = \{r_1, r_3, r_4, r_5, r_6\}$

| | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $\lambda_4$ | $\lambda_5$ | | |
|---|---|---|---|---|---|---|---|
| $r_1$ | 1 | 1 | | | | $\leq$ | 1 |
| $r_2$ | 1 | | 1 | | | $\leq$ | 1 |
| $r_3$ | | 1 | 1 | | | $\leq$ | 1 |
| $r_4$ | 1 | | | | 1 | $\leq$ | 1 |
| $r_5$ | | | 1 | 1 | | $\leq$ | 1 |
| $r_6$ | | | | 1 | 1 | $\leq$ | 1 |

Set packing problem $A\lambda \leq 1$.




Corresponding conflict graph.

*with the set of rows $M$ and columns $P$, and a $|M| \times |P|$ binary coefficient matrix $A$. The SR inequality is defined as:*

$$\sum_{p \in P} \left\lfloor \frac{1}{k} \sum_{i \in S} \alpha_{ip} \right\rfloor \lambda_p \leq \left\lfloor \frac{|S|}{k} \right\rfloor \tag{2.21}$$

*where $S \subseteq M$ and $0 < k \leq |S|$.*

Example 2.3 illustrates some SR inequalities derived from the conflict graph of a set packing problem.

Given a column $p \in P$ we need to have $\sum_{i \in S} \alpha_{ip} \geq k$ to get a non-zero coefficient of $\lambda_p$ in (2.21). For the master problem of VRPTW the coefficient matrix can be translated as $\alpha_{ip} = \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp}$, i.e., $\alpha_{ip}$ is the sum of all the outgoing edges of a customer $i$. Hence,

$$\left\lfloor \frac{1}{k} \sum_{i \in S} \alpha_{ip} \right\rfloor = \left\lfloor \frac{1}{k} \sum_{i \in S} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp} \right\rfloor$$

which is only 1 or larger when $k$ or more customers of $S$ are visited on route $p$.

**Proposition 1.** *The SR inequalities* (2.21) *are valid for the Set Packing structure* $X$*.*

*Proof.* The proof follows directly from Chavtal-Gomory's procedure to construct valid inequalities (Wolsey [36]). Scale the $|S|$ inequalities $\sum_{p\in P} \alpha_{ip}\lambda_p \leq 1$ for each row $i \in S \subseteq M$ from (2.20) with $\frac{1}{k} \geq 0$ and add them:

$$\sum_{p\in P} \frac{1}{k} \sum_{i\in S} \alpha_{ip}\lambda_p \leq \frac{|S|}{k}$$

Flooring on left side and right side leads to (2.21). □

Observe that, when the coefficient $\left\lfloor \frac{1}{k} \sum_{i\in S} \alpha_{ip} \right\rfloor$ evaluates to 0 or 1 for all $p \in P$ and the right hand side $\left\lfloor \frac{|S|}{k} \right\rfloor = 1$ then the set of SR inequalities (2.21) is a subset of the clique inequalities (2.18).

From Definition 1 it is clear that the SR inequalities are Chvatal-Gomory rank-1 cuts, see atal [2]. Eisenbrand [15] has shown that the separation problem is $\mathcal{NP}$-complete for general Chvatal-Gomory rank-1 cuts. However, in some special cases polynomial time separation is possible, e.g. the maximally violated mod-$k$ cuts for a fixed $k$ by Caprara et al. [6]. Since the SR inequalities are another special case, the separation problem will be investigated further.

### 2.3.1 Separation of Subset-Row Inequalities

The separation problem of SR inequalities is defined as follows: Given the current LP-solution $\lambda$ where $\lambda_p < 1$ for all $p \in P$, and let $n$ be the size of $S$. For some fixed values $n$ and $k$ where $1 < k \leq n$, find the most violated SR inequality. Using the binary variable $x_i$ to denote whether $i \in S$ this can be stated as:

$$\max \sum_{p\in P} \left\lfloor \frac{1}{k} \sum_{i\in M} a_{ip}x_i \right\rfloor \lambda_p - \left\lfloor \frac{n}{k} \right\rfloor \tag{2.22}$$

$$\text{s.t.} \sum_{i\in M} x_i = n \tag{2.23}$$

$$x_i \in \{0,1\} \qquad \forall i \in M \tag{2.24}$$

The corresponding decision problem SR-DECISION asks whether

$$\sum_{p\in P} \left\lfloor \frac{1}{k} \sum_{i\in M} a_{ip}x_i \right\rfloor \lambda_p \geq c \tag{2.25}$$

is feasible subject to (2.23) and (2.24), where $1 \leq c < n$ and $c \in \mathbb{Z}$. Since we may multiply (2.25) by any coefficient $\frac{1}{\gamma} > 0$, the coefficient bounds $\lambda_p < 1$

and $c < n$ can be softened to

$$\lambda_p < \frac{1}{\gamma}, \qquad c < \frac{n}{\gamma} \tag{2.26}$$

This leads to the following proposition:

**Proposition 2.** *The separation problem SR-DECISION is $\mathcal{NP}$-complete.*

*Proof.* We will show the statement by reduction from 3-conjunctive normal form satisfiability (3CNF-SAT). Given an expression $\phi$ written in three-conjunctive normal form, the 3CNF-SAT problem asks whether there is an assignment of binary values to the variables such that $\phi$ evaluates to true. An expression is in three-conjunctive normal form when it consists of a collection of disjunctive clauses $C_1, \ldots, C_m$ of literals, where a literal is a variable $x_i$ or a negated variable $\neg x_i$, and each clause contains exactly three literals.

Let $x_1, \ldots, x_n$ be the set of variables which occurs in the clause $\phi$. We transform the 3CNF-SAT instance to a SR-DECISION instance by constructing a matrix $A = (a_{ij})$ with $2n + 3$ rows and $m + n + 1$ columns, i.e., $M = \{1, \ldots, 2n + 3\}$ and $P = \{1, \ldots, m + n + 1\}$.

The rows $1, \ldots, 2n$ of matrix $A$ corresponds to literals $x_1, \neg x_1, x_2, \neg x_2, \ldots, x_n, \neg x_n$, while columns $j = 1, \ldots, m$ correspond to clauses $C_1, \ldots, C_m$, and columns $j = m + 1, \ldots, m + n$ correspond to variables $x_1, \ldots, x_n$.

We now define matrix $A$ as follows: For $j = 1, \ldots, m$ let $a_{ij} = 1$ iff the corresponding literal appears in clause $C_j$. For $j = 1, \ldots, n$ let $a_{i,j+m} = 1$ iff the corresponding literal is $x_j$ or $\neg x_j$. For $j = m+n+1$ let $a_{ij} = 0$. The last three rows of $A$ are defined as follows: For $j = 1, \ldots, m + n$ let $a_{2n+1,j} = 0$, while $a_{2n+1,m+n+1} = 1$. For $j = 1, \ldots, m + n + 1$ let $a_{2n+2,j} = a_{2n+3,j} = 1$. Finally we set $k = 3$, $\lambda_p = 1$ for all $p \in P$ and $c = m + n + 1$. Note that all coefficients are within the bounds (2.26) for $\gamma$ sufficiently large. An example of the transformation is illustrated in Example 2.3.1.

With the chosen constants, the SR-DECISION problem (2.25) reads

$$\sum_{p \in P} \left\lfloor \frac{1}{3} \sum_{i \in M} a_{ip} x_i \right\rfloor \geq m + n + 1 = |P|$$

which is satisfied if and only if

$$\sum_{i \in M} a_{ip} x_i \geq 3 \qquad \forall p \in P$$

As the last three rows of $A$ always must be chosen, it is equivalent to

$$\sum_{i=1}^{2n} a_{ip} x_i \geq 1 \qquad \forall p = 1, \ldots, m + n$$

Illustration of the transformation 3CNF-SAT to SR-DECISION. Given the 3CNF-SAT expression

$$\phi = (x_1 \vee \neg x_1 \vee \neg x_2) \wedge (x_3 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_3 \vee \neg x_4)$$

the matrix $A = (a_{ij})$ becomes

| | | 1 | ... | $m$ | $m+1$ | ... | ... | $m+n$ | $m+n+1$ |
|---|---|---|---|---|---|---|---|---|---|
| | | $C_1$ | ... | $C_m$ | $x_1$ | ... | ... | $x_n$ | |
| 1 | $x_1$ | 1 | | | 1 | | | | |
| 2 | $\neg x_1$ | 1 | | 1 | 1 | | | | |
| | $x_2$ | | 1 | | | 1 | | | |
| $\vdots$ | $\neg x_2$ | 1 | | | | 1 | | | |
| | $x_3$ | | 1 | | | | 1 | | |
| | $\neg x_3$ | | | 1 | | | 1 | | |
| | $x_4$ | | 1 | | | | | 1 | |
| $2n$ | $\neg x_4$ | | | 1 | | | | 1 | |
| $2n+1$ | | | | | | | | | 1 |
| $2n+2$ | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $2n+3$ | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

while we set $k = 3$, $\lambda_p = 1$ for $p \in P$ and $c = 8$.

(i) Assume that there is a feasible assignment of binary values to $x_1, \ldots, x_n$ such that $\phi$ evaluates to true in the 3CNF-SAT instance. In the corresponding SR-DECISION problem choose row $i$ if and only if the corresponding literal is true in $\phi$. Since exactly $n$ literals are true, we will in this way choose $n$ rows. Since at least one literal is true in each clause, and each column $1, \ldots, m$ corresponds to a clause in $A$ we will get a contribution of at least one in each of these columns. Moreover, since exactly one of $x_i$ and $\neg x_i$ is true in $\phi$ we will get a contribution of exactly one in column $m+1, \ldots, m+n$. Hence, the corresponding SR-DECISION problem is true.

(ii) Assume on the other hand that SR-DECISION is true. Let $P' \subseteq P$ be the set of rows corresponding to the solution. By assumption $|P'| = n$. First we notice that exactly one of the rows corresponding to the literals $x_i$ and $\neg x_i$ is chosen. This follows from the fact that we have $n$ columns $m+1, \ldots, m+n$ which needs to be covered by $n$ rows, and each row covers exactly one column. For each literal in $\phi$ let $x_i$ or $\neg x_i$ be true if the corresponding row was chosen in SR-DECISION. Each variable will be well-defined due to the above argument. Moreover, since the rows $P'$ must cover at least one $a_{pi} = 1$ for each column

To illustrate that the bounds (2.26) indeed are realistic consider the case $k = 3$. Choose $\gamma = \frac{m+n+1}{\beta}$ where $\beta = \frac{n-2}{3}$ or $\beta = \frac{n-1}{3}$ depending on which of the expressions that evaluates to an integral value. The right hand side of (2.25) evaluates to

$$c \cdot \frac{1}{\gamma} = (m + n + 1) \cdot \frac{\beta}{m + n + 1} = \beta$$

where an integral value of $\beta$ gives

$$\beta = \left\lfloor \frac{n}{3} \right\rfloor < n$$

The value of $\lambda$ gives

$$\lambda_p \cdot \frac{1}{\gamma} = 1 \cdot \frac{\beta}{m + n + 1} \leq 1 \qquad \forall p \in P$$

Hence all bounds are valid according to the separation problem (2.22)-(2.24).

---

$j = 1, \ldots, m$, we see that each clause in $\phi$ becomes true.

Since the reduction is polynomial, and SR-DECISION obviously is in $\mathcal{NP}$, we have proved the statement. □

Example 2.3.1 shows that typical separation problems of SR inequalities actually possess the properties assumed in the $\mathcal{NP}$-completeness proof.

## 2.4 Label-Setting Algorithm

When solving the pricing problem, it is noted that finding a route with negative reduced cost corresponds to finding a negative cost path starting and ending at the depot, i.e., an ESPPRC. Our ESPPRC algorithm is based on standard label setting techniques presented by e.g. Beasley and Christofides [4], Dumitrescu [14], Feillet et al. [16], Chabrier [7], Danna and Pape [9]; hence in the following we mainly focus on the dominance criterion used for handling the modifications stemming from the SR inequalities of the master problem.

The ESPPRC can be formally defined as: Given a weighted directed graph $G(V, E)$ with nodes $V$ and edges $E$, and a set of resources $R$. For each edge $(i, j) \in E$ and resource $r \in R$ three parameters are given: A lower limit $a_r(i, j)$ on the accumulation of resource $r$ when traversing edge $(i, j) \in E$; an upper limit $b_r(i, j)$ on the accumulation of resource $r$ when traversing edge $(i, j) \in E$; and finally an amount $c_r(i, j)$ of resource $r$ consumed by traversing edge $(i, j) \in E$. The objective is to find a minimum cost path $p$

from a source node $o \in V$ to a target node $o' \in V$, where the accumulated resources of $p$ satisfy the limits for all resources $r \in R$. Without loss of generality, we assume that the limits must be satisfied at the start of each edge $(i, j)$, i.e., before $c_r(i, j)$ has been consumed.

Remark that equivalent upper and lower limits and consumptions on the nodes can be "pushed" onto the edges, e.g., the ingoing edges of the node.

For the pricing problem of the VRPTW, the resources are demand $d$, time $t$, a binary visit-counter for each customer $v \in C$ and reduced cost $\overline{c}$. Note that also the reduced cost is considered a resource. When considering the pricing problem of the VRPTW, the consumptions and upper and lower limits of the resources at each edge $(i, j)$ in ESPPRC are:

$$\begin{array}{llll}
a_d(i,j) = 0, & b_d(i,j) = D - d_j, & c_d(i,j) = d_j & \forall (i,j) \in E \\
a_t(i,j) = a_i, & b_t(i,j) = b_i, & c_t(i,j) = \tau_{ij} & \forall (i,j) \in E \\
a_v(i,j) = 0, & b_v(i,j) = 1, & c_v(i,j) = 1 & \forall v \in V : v = j, \ \forall (i,j) \in E \\
a_v(i,j) = 0, & b_v(i,j) = 1, & c_v(i,j) = 0 & \forall v \in V : v \neq j, \ \forall (i,j) \in E \\
a_{\overline{c}}(i,j) = -\infty, & b_{\overline{c}}(i,j) = \infty\ , & c_{\overline{c}}(i,j) = \overline{c}_{ij} & \forall (i,j) \in E
\end{array}$$

In the label-setting algorithm labels at node $v$ represent partial paths from $o$ to $v$. The following attributes for a label $L$ are considered:

| | |
|---|---|
| $\overline{v}(L)$ | The current end-node of the partial path represented by $L$. |
| $\overline{c}(L)$ | The sum of the reduced cost along path $L$. |
| $r(L)$ | The accumulated consumption of resource $r \in R$ along path $L$. |

A feasible extension $\epsilon \in \mathcal{E}(L)$ of a label $L$ is a partial path starting in a node $\overline{v}(L) \in V$ and ending in the target node $o'$, that does not violate any resources when concatenated with the partial path represented by $L$.

In the following it is assumed that all resources are bounded strongly from above, and weakly from below. This means that if the current resource accumulation of a label is below the lower limit on a given edge, it is allowed to fill up the resource to the lower limit, e.g., waiting for a time window to open. This means that two consecutive labels $L_u$ and $L_v$ related by an edge $(u, v)$, i.e., $L_u$ is extended and creates $L_v$, where $\overline{v}(L_u) = u$ and $\overline{v}(L_v) = v$, must satisfy

$$r(L_v) \leq b_r(u, v), \qquad \forall r \in R \tag{2.27}$$

$$r(L_v) = \max\{r(L_u) + c_r(u, v), a_r(u, v)\}, \qquad \forall r \in R \tag{2.28}$$

Here (2.27) demands that each label $L_u$ satisfies the upper limit $b_r(u, v)$ of resource $r$ corresponding to edge $(u, v)$, while (2.28) states that resource $r$ at label $L_v$ corresponds to the resource consumption at label $L_u$ plus the amount consumed by traversing edge $(u, v)$, respecting the lower limit $a_r(u, v)$ on edge $(u, v)$.

A simple enumeration algorithm could be used to produce all these labels, but to limit the number of labels considered, dominance rules are

introduced to fathom labels which do not lead to an optimal solution.

**Definition 2.** *A label $L_i$ dominates label $L_j$ if*

$$\overline{v}(L_i) = \overline{v}(L_j) \tag{2.29}$$
$$\overline{c}(L_i) \leq \overline{c}(L_j) \tag{2.30}$$
$$\mathcal{E}(L_j) \subseteq \mathcal{E}(L_i) \tag{2.31}$$

In other words, the paths corresponding to labels $L_i$ and $L_j$ should end at the same node $\overline{v}(L_i) = \overline{v}(L_j) \in V$, the path corresponding to label $L_i$ should cost no more than the path corresponding to label $L_j$, and finally any feasible extension of $L_j$ is also a feasible extension of $L_i$.

Feillet et al. [16] suggested to consider the set of nodes that cannot be reached from a label $L_i$ and compare the set with the unreachable nodes of a label $L_j$ in order to determine if some extensions are impossible. Or in other words: update the node resources in an eager fashion instead of a lazy. The following definition is a generalization of Definition 3 in Feillet et al. [16].

**Definition 3.** *Given a start node $o \in V$, a label $L$, and a node $u \in V$ where $\overline{v}(L) = u$ a node $v \in V$ is considered* unreachable *if $v$ has already been visited on the path from $o$ to $u$ or if a resource window is violated, e.g.:*

$$\exists r \in R \qquad\qquad r(L) + \ell_r(u, v) > b_r(v)$$

*where $\ell_r(u, v)$ is a lower bound on the consumption of resource $r$ on all feasible paths from $u$ to $v$. The* node resources *are then given as: $v(L) = 1$ indicates that node $v \in V$ is unreachable from node $\overline{v}(L) \in V$, and $v(L) = 0$ otherwise.*

Determining if (2.31) holds can be quite cumbersome because the straightforward definition demands that we calculate all extensions of the two labels. Therefore, a sufficient criterion for (2.31) is sought that can be computed faster. If label $L_i$ has consumed less resources than label $L_j$ then no resources are limiting the possibilities of extending $L_i$ compared to $L_j$, hence the following proposition can be used as a relaxed version of the dominance criterion.

**Proposition 3.** Desaulniers et al. [11]. *If all resource extension functions are non-decreasing, then label $L_i$ dominates label $L_j$ if:*

$$\overline{v}(L_i) = \overline{v}(L_j) \tag{2.32}$$
$$\overline{c}(L_i) \leq \overline{c}(L_j) \tag{2.33}$$
$$r(L_i) \leq r(L_j) \qquad\qquad \forall r \in R \tag{2.34}$$

Using Proposition 3 as a dominance criterion is a relaxation of the dominance criterion of Definition 2 since only a subset of labels satisfying (2.29), (2.30) and (2.31) satisfies inequalities (2.32), (2.33) and (2.34).

### 2.4.1 Solving the Modified Pricing Problem

Consider some valid SR inequality of the form (2.21),

$$\sum_{p \in P} \left\lfloor \frac{1}{k} \sum_{i \in S} \alpha_{ip} \right\rfloor \lambda_p \leq \left\lfloor \frac{|S|}{k} \right\rfloor$$

where $S \subseteq M$ and $0 < k \leq |S|$. Let $\sigma \leq 0$ be the corresponding dual variable when solving the master problem to LP-optimality. From (2.17) the reduced cost of a column in the VRPTW master problem is:

$$\hat{c}_p = \overline{c}_p - \sigma \left\lfloor \frac{\sum_{i \in S} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp}}{k} \right\rfloor = \sum_{(i,j) \in E} \overline{c}_{ij} \alpha_{ijp} - \sigma \left\lfloor \frac{\sum_{i \in S} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp}}{k} \right\rfloor \tag{2.35}$$

We analyze how this additional cost can be handled in the label-setting algorithm for ESPPRC.

Let $V(L)$ be the nodes visited on the partial path of label $L$. The cost of a label $L$ can then be expressed as:

$$\hat{c}(L) = \overline{c}(L) - \sigma \left\lfloor \frac{|S \cap V(L)|}{k} \right\rfloor \tag{2.36}$$

A new resource $m$ can be used to compute the coefficient of penalty $\sigma$ for label $L$, i.e., $m(L) = |S \cap V(L)|$, the number of customers involved in the cut. Note that the consumption of resource $m$ is 1 for each e.g. outgoing edge of the involved customers. Therefore the usual dominance criterion of Proposition 3 can be used. Note that in case $L_i$ dominates $L_j$, $\overline{c}(L_i) \leq \overline{c}(L_j)$ and $m(L_i) \leq m(L_j)$ so $\hat{c}(L_i) \leq \hat{c}(L_j)$ since $-\sigma > 0$. Hence the penalty term must only be considered on the last edge to the target node to compute the reduced cost $\hat{c}(L)$ of path $L$. However, further labels can be eliminated by exploiting the structure of (2.36).

For a label $L$ let

$$\mathcal{T}(L) = |S \cap V(L)| \bmod k$$

be the number of visits made to $S$ since the last penalty was paid for visiting $k$ nodes in $S$. Recall $\mathcal{E}(L)$ as the set of feasible extensions from the label $L$ to the target node $o'$ and note that when label $L_i$ dominates label $L_j$, their common extensions are $\mathcal{E}(L_j)$ due to (2.31). The following cost dominance criterion is obtained for a single SR inequality:

**Proposition 4.** *If $\mathcal{T}(L_i) \leq \mathcal{T}(L_j)$, $\overline{v}(L_i) = \overline{v}(L_j)$, $\hat{c}(L_i) \leq \hat{c}(L_j)$, and $r(L_i) \leq r(L_j)$ $\forall r \in R$, then label $L_i$ dominates label $L_j$.*

*Proof.* Consider any common extension $\epsilon \in \mathcal{E}(L_j)$. Since $\mathcal{T}(L_i) \leq \mathcal{T}(L_j)$ the relation between the number of future penalties for the two labels when

concatenated with $\epsilon$ is:

$$\left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_i)}{k} \right\rfloor \leq \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_j)}{k} \right\rfloor$$

This leads to the following relation between the costs:

$$\begin{aligned} & \hat{c}(L_i + \epsilon) = \hat{c}(L_i) + \overline{c}(\epsilon) - \sigma \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_i)}{k} \right\rfloor \\ \leq \quad & \hat{c}(L_j + \epsilon) = \hat{c}(L_j) + \overline{c}(\epsilon) - \sigma \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_j)}{k} \right\rfloor \end{aligned}$$

Hence label $L_i$ dominates label $L_j$. □

**Proposition 5.** *If $\mathcal{T}(L_i) > \mathcal{T}(L_j)$, $\overline{v}(L_i) = \overline{v}(L_j)$, $\hat{c}(L_i) - \sigma \leq \hat{c}(L_j)$, and $r(L_i) \leq r(L_j)\ \forall r \in R$, then label $L_i$ dominates label $L_j$.*

*Proof.* Consider any common extension $\epsilon \in \mathcal{E}(L_j)$. Since $\mathcal{T}(L_i) > \mathcal{T}(L_j)$ the relation between the number of future penalties for the two labels when concatenated with $\epsilon$ is:

$$\left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_i)}{k} \right\rfloor \geq \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_j)}{k} \right\rfloor \tag{2.37}$$

Since $0 \leq \mathcal{T}(L_j) < \mathcal{T}(L_i) \leq k$ it is clear that the left hand side of (2.37) is at most one unit larger than the right hand side, i.e., label $L_i$ will pay the penalty at most one more time than label $L_j$. Hence,

$$\left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_i)}{k} \right\rfloor - 1 \leq \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_j)}{k} \right\rfloor$$

That is, the additional cost of extending $L_i$ with $\epsilon$ is at most $-\sigma$ more than extending $L_j$ with $\epsilon$. This leads to the following relation between the costs:

$$\begin{aligned} \hat{c}(L_i + \epsilon) &= \hat{c}(L_i) + \overline{c}(\epsilon) - \sigma \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_i)}{k} \right\rfloor \\ &= \hat{c}(L_i) - \sigma + \overline{c}(\epsilon) - \sigma \left( \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_i)}{k} \right\rfloor - 1 \right) \\ &\leq \hat{c}(L_j) + \overline{c}(\epsilon) - \sigma \left\lfloor \frac{|S \cap \epsilon| + \mathcal{T}(L_j)}{k} \right\rfloor \\ &= \hat{c}(L_j + \epsilon) \end{aligned}$$

Hence label $L_i$ dominates label $L_j$. □

Observe that if $\mathcal{T}(L_i) + |S \cap \epsilon| < k$ for all $\epsilon \in \mathcal{E}(L_j)$, it is not possible to visit $S$ enough times to trigger a penalty, i.e., the temporary penalty to the cost of $L_i$ can be disregarded.

In case of several SR inequalities, the new dominance criterion is as follows:

**Proposition 6.** *Let* $Q = \{q : \sigma_q < 0 \wedge \mathcal{T}_q(L_i) > \mathcal{T}_q(L_j)\}$. *Then label* $L_i$ *dominates label* $L_j$ *if:*

$$\overline{v}(L_i) = \overline{v}(L_j) \tag{2.38}$$

$$\hat{c}(L_i) - \sum_{q \in Q} \sigma_q \leq \hat{c}(L_j) \tag{2.39}$$

$$r(L_i) \leq r(L_j) \qquad \forall r \in R \tag{2.40}$$

*Proof.* The validity of (2.39) follows directly from Propositions 4 and 5. The validity of (2.38) and (2.40) follows from Proposition 3. □

## 2.5 Computational Results

The BCP algorithm has been implemented using the BCP framework and the open source linear programming solver CLP, both parts of the framework coi [1]. All tests are run on an Intel® Pentium® 4 3.0 GHz PC with 4 GB of memory.

The benchmarks of Solomon [33] follow a naming convention of `DTm.n`. The distribution `D` can be R, C and RC, where the C instances have a clustered distribution of customers, the R instances have a random distribution of customers, and the RC instances are a mix of clustered and randomly distributed customers. The time window `T` is either 1 or 2, where instances of type 1 have tighter time windows than instances of type 2. The instance number is given by `m` and the number of customers is given by `n`.

The outline of the BCP algorithm presented in this paper is as follows:

*Step 1.* Choose an unprocessed branch node. If the lower bound is above the upper bound, then fathom branch node.

*Step 2.* Solve the LP master problem.

*Step 3.* Solve the pricing problem heuristically. If columns with negative reduced cost have been found, then add them to the master problem and go back to Step 2.

*Step 4.* Solve the pricing problem to optimality. Update the lower bound. If the lower bound is above the upper bound, then fathom the branch node. If some new columns have been found, then add them to the master problem and go to Step 2.

*Step 5.* Separate SR inequalities. If any violated cuts are found, then add them to the master problem and go to Step 2.

*Step 6.* If the LP solution is fractional then branch and add the children to the set of unprocessed branch nodes. Mark the current node as processed and go to Step 1.

We allow a maximum of 400 variables and 50 cuts to be generated in each of steps 3, 4, and 5 respectively. The pricing-problem heuristic is based on the label-setting algorithm but a simpler heuristic dominance criterion

is used. If a label $L_i$ dominates $L_j$ on *cost*, *demand* and *time* it is regarded as dominated and $L_j$ is discarded. That is, no concern is taken to the *node* resources. The separation of SR inequalities is done with a complete enumeration of all inequalities with $|S| = 3$ and $k = 2$. Let $B$ be the set of basic variables in the current LP solution and $C$ be the set of customers, then the separation can be done in $O(|C|^3|B|)$. Preliminary tests showed that SR inequalities with different values of $n$ and $k$ seldom appeared in the VRPTW instances, hence no separation of these inequalities was done.

The branch tree is explored with a best-bound search strategy, i.e., the node with the lowest lower bound is chosen first, breaking ties based on the LP result of the strong branching. We have adapted the branching rule used by Fukasawa et al. [17]: For a subset of customers $S \subset C$ the number of vehicles to visit that set is either two or greater than or equal to four, i.e.,

$$\sum_{k \in K} \sum_{(i,j) \in \delta^+(S)} (x_{ijk} + x_{jik}) = 2$$

and

$$\sum_{k \in K} \sum_{(i,j) \in \delta^+(S)} (x_{ijk} + x_{jik}) \geq 4$$

We are using the cut library of Lysgaard [26] to separate candidate sets for branching, which is an implementation of the heuristic methods described in Lysgaard et al. [27].

| Author(s) | CPU | SpecINT | SpecCFP | Normalized |
|---|---|---|---|---|
| Irnich and Villeneuve [20] | P3 600 MHz* | 295 | 204 | 0.23 |
| Chabrier [7] | P4 1.5 GHz | 526 | 606 | 0.52 |
| Jepsen et al. [this paper] | P4 3.0 GHz | 1099 | 1077 | 1.00 |

Table 2.1: Comparison of computer speed. Based on CPU2000 benchmarks from SPEC [34]. (*) benchmarks are given for P3 650 MHz since no benchmarks were available for P3 600. The normalized value is an average of SpecINT and SpecCFP.

### 2.5.1 Running Times

To give a fair comparison between running times of our algorithm and the two most recent algorithms presented by Irnich and Villeneuve [20] and Chabrier [7], the CPU speed is taken into account. This is done according to the CPU2000 benchmarks reported by The Standard Performance Evaluation Corporation SPEC [34]. Table 2.1 gives the integer and floating point benchmark scores and a normalized value, e.g. our computations were carried out on a computer approximately twice as fast as that of Chabrier.

A comparison of running times is shown in Table 2.2. To save space we only report results on what we consider hard instances, i.e., the Solomon

instances that were closed by either Irnich and Villeneuve [20] or Chabrier [7] and by us.

Our algorithm outperforms those of Irnich and Villeneuve and Chabrier for 17 out of 22 instances. Seven of these instances were solved without any SR inequalities. In these cases, the faster running times were probably due to the bi-directional label-setting algorithm.

With the introduction of SR inequalities our algorithm becomes competitive with the algorithm based on solving $k$-cyc-SPPRC (e.g. instances R104.100, RC104.100, RC107.100, RC108.100, and R211.50) and clearly outperforms the ESPPRC based algorithm on the harder instances (e.g., instances R210.50, RC202.100, RC205.100, and RC208.25). In some cases when solving the C1 and C2 instances the BCP algorithm tails off leading to slow solution times or no solution at all. However, this must be seen in the light of a simple implementation and no use of other cutting planes than the SR inequalities.

### 2.5.2 Comparing Lower Bounds in the Root Node

Table 2.3 reports the lower bounds obtained in the root node of the master problem with and without SR inequalities and with best bounds obtained by Irnich and Villeneuve [20] using $k$-cyc-SPPRC. Again we only report results on what we consider the hard instances from Table 2.2 plus the instances closed by us.

As seen, the lower bounds obtained with SR inequalities are improved quite significantly for most of the instances. Moreover, in most cases the problems are solved without branching. Out of the 32 instances considered, the gap was closed in the root node in 8 instances due to the ESPPRC and in an additional 16 instances due to the SR inequalities. However, one needs to take into account that the running time of solving the root node is increased due to the increased difficulty of the pricing problems.

| | Irnich and Villeneuve [20] | Chabrier [7] | Jepsen et al. [this paper] | | | |
|---|---|---|---|---|---|---|
| Instance | Time (s) | Time (s) | Time (s) | Speedup | | |
| R104.100 | 268106.0 | - | **32343.9** | 1.9 | / | - |
| RC104.100 | 986809.0 | - | **65806.8** | 3.4 | / | - |
| RC107.100 | 42770.7 | - | **153.8** | 64.0 | / | - |
| RC108.100 | 71263.0 | - | **3365.0** | 4.9 | / | - |
| R203.50 | **217.1** | 3320.9 | 50.8 | 1.0 | / | 34.0 |
| R204.25 | 123.1 | 171.6 | **7.5** | 3.8 | / | 11.9 |
| R205.50 | 585.7 | 531.0 | **15.5** | 8.6 | / | 17.8 |
| R206.50 | 22455.3 | 4656.1 | **190.9** | 27.1 | / | 12.7 |
| R208.25 | 321.9 | 741.5 | *__2.9__ | 25.5 | / | 133.0 |
| R209.50 | 142.4 | 195.4 | **16.6** | 2.0 | / | 6.1 |
| R210.50 | 11551.4 | 65638.6 | *__332.7__ | 8.0 | / | 102.6 |
| R211.50 | **21323.0** | - | 10543.8 | 0.5 | / | - |
| RC202.50 | 241.6 | **13.0** | *10.7 | 5.2 | / | 0.6 |
| RC202.100 | 124018.0 | 19636.5 | **312.6** | 91.2 | / | 32.7 |
| RC203.25 | 1876.0 | 5.1 | *__0.7__ | 616.4 | / | 3.8 |
| RC203.50 | 54229.2 | 4481.5 | *__190.9__ | 65.3 | / | 12.2 |
| RC204.25 | - | 13.0 | *__2.0__ | - | / | 3.4 |
| RC205.50 | 52.6 | **10.6** | *5.9 | 2.1 | / | 0.9 |
| RC205.100 | 13295.9 | 15151.7 | **221.2** | 13.8 | / | 35.6 |
| RC206.50 | 469.1 | **9.4** | *8.2 | 13.2 | / | 0.6 |
| RC207.50 | - | 71.1 | *__21.5__ | - | / | 1.7 |
| RC208.25 | - | 33785.3 | **78.4** | - | / | 224.1 |

Table 2.2: Comparison of running time. Speedup is calculated based on the normalized values in Table 2.1 and are versus Irnich and Villeneuve and Chabrier respectively. Results with (*) are based on an algorithm without the SR inequalities. Results in **boldface** indicate the fastest algorithm after normalization. (-) indicates that no running times were provided by the author(s) or that the instance was not solved.

| | | Irnich and Villeneuve [20] | | Jepsen et al. [this paper] | |
|---|---|---|---|---|---|
| Instance | UB | $k$ | LB | LB(1) | LB(2) |
| R104.100 | 971.5 | 3 | 955.8 | 956.9 | 971.3 |
| **R108.100** | 932.1 | 4 | 913.9 | 913.6 | **932.1** |
| **R112.100** | 948.6 | 3 | 925.9 | 926.8 | 946.7 |
| RC104.100 | 1132.3 | 3 | 1114.4 | 1101.9 | 1129.9 |
| RC106.100 | 1372.7 | 4 | 1343.1 | 1318.8 | 1367.3 |
| RC107.100 | 1207.8 | 4 | 1195.4 | 1183.4 | **1207.8** |
| RC108.100 | 1114.2 | 3 | 1100.5 | 1073.5 | **1114.2** |
| **R202.100** | 1029.6 | 0 | 933.5 | 1022.3 | 1027.3 |
| R203.50 | 605.3 | 4 | 598.6 | 598.6 | **605.3** |
| **R203.100** | 870.8 | 2 | 847.1 | 867.0 | **870.8** |
| R204.25 | 355.0 | 4 | 349.1 | 350.5 | **355.0** |
| R205.50 | 690.1 | 4 | 682.8 | 682.9 | **690.1** |
| R206.50 | 632.4 | 4 | 621.3 | 626.4 | **632.4** |
| **R207.50** | 575.5 | 4 | 557.4 | 564.1 | **575.5** |
| R208.25 | 328.2 | 4 | 327.1 | **328.2** | **328.2** |
| R209.50 | 600.6 | 4 | 599.9 | 599.9 | **600.6** |
| **R209.100** | 854.8 | 3 | 834.4 | 841.5 | 854.4 |
| R210.50 | 645.6 | 4 | 633.1 | 636.1 | 645.3 |
| R211.50 | 535.5 | 4 | 526.0 | 528.7 | **535.5** |
| RC202.50 | 613.6 | 4 | 604.5 | **613.6** | **613.6** |
| RC202.100 | 1092.3 | 3 | 1055.0 | 1088.1 | **1092.3** |
| RC203.25 | 326.9 | 4 | 297.7 | **326.9** | **326.9** |
| RC203.50 | 555.3 | 4 | 530.0 | **555.3** | **555.3** |
| **RC203.100** | 923.7 | 0 | 693.7 | 922.6 | **923.7** |
| RC204.25 | 299.7 | 4 | 266.3 | **299.7** | **299.7** |
| RC205.50 | 630.2 | 4 | **630.2** | **630.2** | **630.2** |
| RC205.100 | 1154.0 | 3 | 1130.5 | 1147.7 | **1154.0** |
| RC206.50 | 610.0 | 4 | 597.1 | **610.0** | **610.0** |
| **RC206.100** | 1051.1 | 3 | 1017.0 | 1038.6 | **1051.1** |
| RC207.50 | 558.6 | 4 | 504.9 | **558.6** | **558.6** |
| RC208.25 | 269.1 | 4 | 238.3 | **269.1** | **269.1** |
| RC208.50 | 476.7 | 3 | 422.3 | 472.3 | **476.7** |

Table 2.3: Comparison of root lower bounds. LB by Irnich and Villeneuve is the best lower bound obtained with $k$-cyc-SPPRC and valid inequalities, LB(1) is with ESPPRC and LB(2) is with ESPPRC and SR inequalities. Lower bounds in **boldface** indicate lower bounds equal to the upper bound. Instances in **boldface** are the Solomon instances closed by us.

| | | 25 customers | | 50 customers | | 100 customers | |
|---|---|---|---|---|---|---|---|
| Class | No. | Prev. | Jepsen et al. [this paper] | Prev. | Jepsen et al. [this paper] | Prev | Jepsen et al. [this paper] |
| R1 | 12 | 12 | 12 | 12 | 12 | 10 | 12 |
| C1 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| RC1 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| R2 | 11 | 11 | 11 | 9 | 9 | 1 | 4 |
| C2 | 8 | 8 | 8 | 8 | 7 | 8 | 7 |
| RC2 | 8 | 8 | 8 | 8 | 7 | 3 | 5 |
| Summary | 56 | 56 | 56 | 55 | 52 | 39 | 45 |

Table 2.4: Summary of solved Solomon instances. No. is the number of instances in that class, and for 25, 50 and 100 customers the two columns refers to the number of instances previously solved to optimality and the number of instances solved to optimality by us.

| Instance | UB | LB | M | B | LP | $\text{Time}_\text{r}$(s) | $\text{Time}_\text{var}$(s) | $\text{Time}_\text{LP}$(s) | Time (s) |
|---|---|---|---|---|---|---|---|---|---|
| R108.100 | 932.1 | 932.1 | 10 | 1 | 132 | 5911.71 | 5796.04 | 77.36 | 5911.74 |
| R112.100 | 948.6 | 946.7 | 10 | 9 | 351 | 55573.68 | 199907.03 | 1598.63 | 202803.94 |
| R202.100 | 1029.6 | 1027.3 | 8 | 13 | 514 | 974.51 | 730.04 | 4810.47 | 8282.38 |
| R203.100 | 870.8 | 870.8 | 6 | 1 | 447 | 54187.15 | 48474.45 | 3973.42 | 54187.40 |
| R207.50 | 575.5 | 575.5 | 3 | 1 | 107 | 34406.92 | 34282.47 | 118.69 | 34406.96 |
| R209.100 | 854.8 | 854.4 | 5 | 3 | 337 | 31547.45 | 74779.58 | 2978.42 | 78560.47 |
| RC203.100 | 923.7 | 923.7 | 5 | 1 | 402 | 14917.18 | 13873.53 | 1025.65 | 14917.36 |
| RC206.100 | 1051.1 | 1051.1 | 7 | 1 | 179 | 339.63 | 159.33 | 171.34 | 339.69 |

Table 2.5: Instances closed by Jepsen et al. [this paper]. *UB* is the optimal solution found by us, *LB* is lower bound at the root node, *M* is the number of vehicles in the solution, *B* is the number of branch nodes, *LP* is the number of LP iterations, $Time_r$ is the time solving the root node, $Time_{var}$ is time spent solving the pricing problem, $Time_{LP}$ is the time spent solving LP problems, and *Time* is the total time.

### 2.5.3 Closed Solomon Instances

Table 2.4 gives an overview of how many instances were solved for each class of the Solomon instances. We were able to close 8 previously unsolved instances. We did not succeed to solve four previously solved instances (R204.50, C204.50, C204.100, and RC204.50).

Information on all solved Solomon instances can be found in Tables A.6–A.8 in Appendix A.1. Furthermore Table 2.5 provides detailed information of the instances closed in this paper. The solutions can be found in Tables A.9–A.16 in Appendix A.2.

## 2.6 Concluding Remarks

The introduction of the SR inequalities significantly improved the results of the BCP algorithm. This made it possible to solve 8 previously unsolved instances from the Solomon benchmarks.

Except for four cases (R204.50, C204.50 and C204.100 solved with $k$-cyc-SPPRC by Irnich and Villeneuve [20] and RC204.50 solved by Danna and Pape [9]) our BCP algorithm is competitive and in most cases superior to earlier algorithms within this field. With minor modifications in the definition of the conflict graph the SR inequalities can be applied to the $k$-cyc-SPPRC algorithm using the same cost-modified dominance criterion as described in this paper. Preliminary results by Jepsen et al. [21] have shown that the lower bounds obtained in a BCP algorithm for VRPTW using the $k$-cyc-SPPRC algorithm and SR inequalities are almost as good as those obtained using the approach presented in this paper. This seems to be a promising direction of research in order to solve large VRPTW instances, since the ESPPRC algorithm is considerably slower than the $k$-cyc-SPPRC algorithm when the number of customers increases.

Moreover, we note that the SR inequalities can be applied to any set packing problem. That is, they can be used in BCP algorithms for other problems with a set packing problem master problem. One only needs to consider how the dual variables of the SR inequalities are handled in the pricing problems, however this is not necessarily trivial and must be investigated for the individual pricing problems.

Adding SR inequalities to the master problem means that the pricing problem becomes a shortest path problem with non-additive non-decreasing constraints or objective function. By modifying the dominance criterion, we have shown that this is tractable in a label-setting algorithm. A further discussion of shortest path problems with various non-additive constraints can be found in Pisinger and Reinhardt [30]. The development of algorithms which efficiently handle non-additive constraints is important to increase the number of valid inequalities which can be handled.

## A.1 Results on Solomon Instances

This appendix contains detailed information about solved Solomon instances. The first column of the tables is the instance name, then three columns for the branch-and-cut-and-price algorithm with ESPPRC and with ESPPRC and SR-inequalities follow. The columns are the lower bound in the root node, the number of branch tree nodes and the total running time. A (-) means that the instance was not solved. The last two columns are the optimal upper bound and a reference to the authors who were the first to solve that instance, disregarding Desrochers et al. [10] who

solved many of the instances with a different calculation of the travel times making it hard to compare with later solutions. The author legend is:

| | |
|---|---|
| C: | Chabrier [7] |
| CR: | Cook and Rich [8] |
| DLP: | Danna and Pape [9] |
| IV: | Irnich and Villeneuve [20] |
| JPSP: | Jepsen et al. [this paper] |
| KDMSS: | Kohl et al. [23] |
| KLM: | Kallehauge et al. [22] |
| L: | Larsen [24] |
| S: | Salani [32] |

| | with ESPPRC | | | with ESPPRC and SR | | | | |
|---|---|---|---|---|---|---|---|---|
| Instance | LB | Tree | Time (s) | LB | Tree | Time (s) | UB | Ref. |
| R101 | 617.1 | 1 | 0.02 | 617.1 | 1 | 0.02 | 617.1 | KDMSS |
| R102 | 546.4 | 3 | 0.13 | 547.1 | 1 | 0.09 | 547.1 | KDMSS |
| R103 | 454.6 | 1 | 0.11 | 454.6 | 1 | 0.11 | 454.6 | KDMSS |
| R104 | 416.9 | 1 | 0.12 | 416.9 | 1 | 0.12 | 416.9 | KDMSS |
| R105 | 530.5 | 1 | 0.02 | 530.5 | 1 | 0.02 | 530.5 | KDMSS |
| R106 | 457.3 | 5 | 0.29 | 465.4 | 1 | 0.10 | 465.4 | KDMSS |
| R107 | 424.3 | 1 | 0.12 | 424.3 | 1 | 0.12 | 424.3 | KDMSS |
| R108 | 396.9 | 3 | 0.31 | 397.3 | 1 | 0.24 | 397.3 | KDMSS |
| R109 | 441.3 | 1 | 0.06 | 441.3 | 1 | 0.06 | 441.3 | KDMSS |
| R110 | 438.4 | 17 | 1.16 | 444.1 | 3 | 0.29 | 444.1 | KDMSS |
| R111 | 427.3 | 3 | 0.23 | 428.8 | 1 | 0.13 | 428.8 | KDMSS |
| R112 | 387.1 | 13 | 1.19 | 393.0 | 1 | 0.52 | 393.0 | KDMSS |
| C101 | 191.3 | 1 | 0.13 | 191.3 | 1 | 0.13 | 191.3 | KDMSS |
| C102 | 190.3 | 1 | 0.53 | 190.3 | 1 | 0.53 | 190.3 | KDMSS |
| C103 | 190.3 | 1 | 0.80 | 190.3 | 1 | 0.80 | 190.3 | KDMSS |
| C104 | 186.9 | 1 | 3.29 | 186.9 | 1 | 3.29 | 186.9 | KDMSS |
| C105 | 191.3 | 1 | 0.17 | 191.3 | 1 | 0.17 | 191.3 | KDMSS |
| C106 | 191.3 | 1 | 0.14 | 191.3 | 1 | 0.14 | 191.3 | KDMSS |
| C107 | 191.3 | 1 | 0.20 | 191.3 | 1 | 0.20 | 191.3 | KDMSS |
| C108 | 191.3 | 1 | 0.37 | 191.3 | 1 | 0.37 | 191.3 | KDMSS |
| C109 | 191.3 | 1 | 0.62 | 191.3 | 1 | 0.62 | 191.3 | KDMSS |
| RC101 | 406.7 | 5 | 0.20 | 461.1 | 1 | 0.09 | 461.1 | KDMSS |
| RC102 | 351.8 | 1 | 0.05 | 351.8 | 1 | 0.05 | 351.8 | KDMSS |
| RC103 | 332.8 | 1 | 0.19 | 332.8 | 1 | 0.19 | 332.8 | KDMSS |
| RC104 | 306.6 | 1 | 0.52 | 306.6 | 1 | 0.52 | 306.6 | KDMSS |
| RC105 | 411.3 | 1 | 0.06 | 411.3 | 1 | 0.06 | 411.3 | KDMSS |
| RC106 | 345.5 | 1 | 0.10 | 345.5 | 1 | 0.10 | 345.5 | KDMSS |
| RC107 | 298.3 | 1 | 0.29 | 298.3 | 1 | 0.29 | 298.3 | KDMSS |
| RC108 | 294.5 | 1 | 0.67 | 294.5 | 1 | 0.67 | 294.5 | KDMSS |
| R201 | 460.1 | 3 | 0.44 | 463.3 | 1 | 0.27 | 463.3 | CR+KLM |
| R202 | 410.5 | 1 | 0.61 | 410.5 | 1 | 0.61 | 410.5 | CR+KLM |
| R203 | 391.4 | 1 | 0.80 | 391.4 | 1 | 0.80 | 391.4 | CR+KLM |
| R204 | 350.5 | 19 | 18.40 | 355.0 | 1 | 7.51 | 355.0 | IV+C |
| R205 | 390.6 | 3 | 1.62 | 393.0 | 1 | 1.06 | 393.0 | CR+KLM |
| R206 | 373.6 | 3 | 1.67 | 374.4 | 1 | 0.93 | 374.4 | CR+KLM |
| R207 | 360.1 | 5 | 4.03 | 361.6 | 1 | 1.39 | 361.6 | KLM |
| R208 | 328.2 | 1 | 2.87 | 328.2 | 1 | 2.87 | 328.2 | IV+C |
| R209 | 364.1 | 9 | 4.99 | 370.7 | 1 | 2.26 | 370.7 | KLM |
| R210 | 404.2 | 3 | 1.52 | 404.6 | 1 | 1.04 | 404.6 | CR+KLM |
| R211 | 341.4 | 29 | 38.17 | 350.9 | 1 | 22.62 | 350.9 | KLM |
| C201 | 214.7 | 1 | 0.84 | 214.7 | 1 | 0.84 | 214.7 | CR+L |
| C202 | 214.7 | 1 | 3.00 | 214.7 | 1 | 3.00 | 214.7 | CR+L |
| C203 | 214.7 | 1 | 3.02 | 214.7 | 1 | 3.02 | 214.7 | CR+L |
| C204 | 213.1 | 1 | 7.00 | 213.1 | 1 | 7.00 | 213.1 | CR+KLM |
| C205 | 214.7 | 1 | 1.10 | 214.7 | 1 | 1.10 | 214.7 | CR+L |
| C206 | 214.7 | 1 | 1.75 | 214.7 | 1 | 1.75 | 214.7 | CR+L |
| C207 | 214.5 | 1 | 2.70 | 214.5 | 1 | 2.70 | 214.5 | CR+L |
| C208 | 214.5 | 1 | 1.85 | 214.5 | 1 | 1.85 | 214.5 | CR+L |
| RC201 | 360.2 | 1 | 0.25 | 360.2 | 1 | 0.25 | 360.2 | CR+L |
| RC202 | 338.0 | 1 | 0.58 | 338.0 | 1 | 0.58 | 338.0 | CR+KLM |
| RC203 | 326.9 | 1 | 0.72 | 326.9 | 1 | 0.72 | 326.9 | IV+C |
| RC204 | 299.7 | 1 | 1.95 | 299.7 | 1 | 1.95 | 299.7 | C |
| RC205 | 338.0 | 1 | 0.62 | 338.0 | 1 | 0.62 | 338.0 | L+KLM |
| RC206 | 324.0 | 1 | 0.87 | 324.0 | 1 | 0.87 | 324.0 | KLM |
| RC207 | 298.3 | 1 | 0.88 | 298.3 | 1 | 0.88 | 298.3 | KLM |
| RC208 | 269.1 | 1 | 78.42 | 269.1 | 1 | 78.42 | 269.1 | C |



Table A.6: Instances with 25 customers.

| | with ESPPRC | | | with ESPPRC and SR | | | | |
|---|---|---|---|---|---|---|---|---|
| Instance | LB | Tree | Time (s) | LB | Tree | Time (s) | UB | Ref. |
| R101 | 1043.4 | 3 | 0.14 | 1044.0 | 1 | 0.09 | 1044.0 | KDMSS |
| R102 | 909.0 | 1 | 0.27 | 909.0 | 1 | 0.27 | 909.0 | KDMSS |
| R103 | 769.3 | 13 | 4.98 | 772.9 | 1 | 2.02 | 772.9 | KDMSS |
| R104 | 619.1 | 21 | 33.29 | 625.4 | 1 | 6.73 | 625.4 | KDMSS |
| R105 | 892.2 | 29 | 2.78 | 893.7 | 5 | 1.15 | 899.3 | KDMSS |
| R106 | 791.4 | 5 | 1.41 | 793.0 | 1 | 0.83 | 793.0 | KDMSS |
| R107 | 707.3 | 11 | 5.56 | 711.1 | 1 | 4.76 | 711.1 | KDMSS |
| R108 | 594.7 | 789 | 1723.29 | 607.4 | 23 | 1601.68 | 617.7 | CR+KLM |
| R109 | 775.4 | 77 | 20.11 | 783.3 | 7 | 11.54 | 786.8 | KDMSS |
| R110 | 695.1 | 9 | 3.38 | 697.0 | 1 | 1.46 | 697.0 | KDMSS |
| R111 | 696.3 | 41 | 19.21 | 707.2 | 1 | 3.67 | 707.2 | CR+KLM |
| R112 | 614.9 | 165 | 169.26 | 630.2 | 1 | 35.67 | 630.2 | CR+KLM |
| C101 | 362.4 | 1 | 0.47 | 362.4 | 1 | 0.47 | 362.4 | KDMSS |
| C102 | 361.4 | 1 | 1.59 | 361.4 | 1 | 1.59 | 361.4 | KDMSS |
| C103 | 361.4 | 1 | 6.06 | 361.4 | 1 | 6.06 | 361.4 | KDMSS |
| C104 | 358.0 | 1 | 1564.88 | 358.0 | 1 | 1564.88 | 358.0 | KDMSS |
| C105 | 362.4 | 1 | 0.49 | 362.4 | 1 | 0.49 | 362.4 | KDMSS |
| C106 | 362.4 | 1 | 0.69 | 362.4 | 1 | 0.69 | 362.4 | KDMSS |
| C107 | 362.4 | 1 | 0.97 | 362.4 | 1 | 0.97 | 362.4 | KDMSS |
| C108 | 362.4 | 1 | 1.55 | 362.4 | 1 | 1.55 | 362.4 | KDMSS |
| C109 | 362.4 | 1 | 3.62 | 362.4 | 1 | 3.62 | 362.4 | KDMSS |
| RC101 | 850.1 | 39 | 5.60 | 944.0 | 1 | 2.12 | 944.0 | KDMSS |
| RC102 | 721.9 | 127 | 60.41 | 822.5 | 1 | 8.68 | 822.5 | KDMSS |
| RC103 | 645.3 | 9 | 8.56 | 710.9 | 1 | 40.05 | 710.9 | KDMSS |
| RC104 | 545.8 | 1 | 5.71 | 545.8 | 1 | 5.71 | 545.8 | KDMSS |
| RC105 | 761.6 | 21 | 7.22 | 855.3 | 1 | 4.31 | 855.3 | KDMSS |
| RC106 | 664.5 | 11 | 3.35 | 723.2 | 1 | 3.88 | 723.2 | KDMSS |
| RC107 | 603.6 | 7 | 4.60 | 642.7 | 1 | 4.49 | 642.7 | KDMSS |
| RC108 | 541.2 | 5 | 15.88 | 594.8 | 5 | 260.95 | 598.1 | KDMSS |
| R201 | 791.9 | 1 | 4.97 | 791.9 | 1 | 4.97 | 791.9 | CR+KLM |
| R202 | 698.5 | 1 | 9.88 | 698.5 | 1 | 9.88 | 698.5 | CR+KLM |
| R203 | 598.6 | 25 | 355.99 | 605.3 | 1 | 50.80 | 605.3 | IV+C |
| R204 | | | - | | | - | 506.4 | IV |
| R205 | 682.9 | 35 | 118.12 | 690.1 | 1 | 15.45 | 690.1 | IV+C |
| R206 | 626.4 | 47 | 288.00 | 632.4 | 1 | 190.86 | 632.4 | IV+C |
| R207 | 564.1 | 141 | 15400.44 | 575.5 | 1 | 34406.96 | 575.5 | **JPSP** |
| R208 | | | - | | | - | - | - |
| R209 | 599.9 | 3 | 24.45 | 600.6 | 1 | 16.63 | 600.6 | IV+C |
| R210 | 636.1 | 49 | 332.70 | 645.3 | 3 | 18545.61 | 645.6 | IV+C |
| R211 | 528.7 | 31 | 44644.89 | 535.5 | 1 | 10543.81 | 535.5 | IV+DLP |
| C201 | 360.2 | 1 | 42.07 | 360.2 | 1 | 42.07 | 360.2 | CR+L |
| C202 | 360.2 | 1 | 67.05 | 360.2 | 1 | 67.05 | 360.2 | CR+KLM |
| C203 | 359.8 | 1 | 214.88 | 359.8 | 1 | 214.88 | 359.8 | CR+KLM |
| C204 | | | - | | | - | 350.1 | KLM |
| C205 | 359.8 | 1 | 64.18 | 359.8 | 1 | 64.18 | 359.8 | CR+KLM |
| C206 | 359.8 | 1 | 38.91 | 359.8 | 1 | 38.91 | 359.8 | CR+KLM |
| C207 | 359.6 | 1 | 72.81 | 359.6 | 1 | 72.81 | 359.6 | CR+KLM |
| C208 | 350.5 | 1 | 55.79 | 350.5 | 1 | 55.79 | 350.5 | CR+KLM |
| RC201 | 684.8 | 1 | 3.00 | 684.8 | 1 | 3.00 | 684.8 | L+KLM |
| RC202 | 613.6 | 1 | 10.69 | 613.6 | 1 | 10.69 | 613.6 | IV+C |
| RC203 | 555.3 | 1 | 190.88 | 555.3 | 1 | 190.88 | 555.3 | IV+C |
| RC204 | | | - | | | - | 442.2 | DLP |
| RC205 | 630.2 | 1 | 5.88 | 630.2 | 1 | 5.88 | 630.2 | IV+C |
| RC206 | 610.0 | 1 | 8.17 | 610.0 | 1 | 8.17 | 610.0 | IV+C |
| RC207 | 558.6 | 1 | 21.53 | 558.6 | 1 | 21.53 | 558.6 | C |
| RC208 | | | - | 476.7 | 1 | 1639.40 | 476.7 | S |



Table A.7: Instances with 50 customers.

| | with ESPPRC | | | with ESPPRC and SR | | | | |
|---|---|---|---|---|---|---|---|---|
| Instance | LB | Tree | Time (s) | LB | Tree | Time (s) | UB | Ref. |
| R101 | 1631.2 | 57 | 20.08 | 1634.0 | 3 | 1.87 | 1637.7 | KDMSS |
| R102 | 1466.6 | 1 | 4.39 | 1466.6 | 1 | 4.39 | 1466.6 | KDMSS |
| R103 | 1206.8 | 19 | 55.78 | 1208.7 | 1 | 23.85 | 1208.7 | CR+L |
| R104 | | | - | 971.3 | 3 | 32343.92 | 971.5 | IV |
| R105 | 1346.2 | 113 | 126.96 | 1355.2 | 5 | 43.12 | 1355.3 | KDMSS |
| R106 | 1227.0 | 147 | 511.07 | 1234.6 | 1 | 75.42 | 1234.6 | CR+KLM |
| R107 | | | - | 1064.3 | 3 | 1310.30 | 1064.6 | CR+KLM |
| R108 | | | - | 932.1 | 1 | 5911.74 | 932.1 | **JPSP** |
| R109 | | | - | 1144.1 | 19 | 1432.41 | 1146.9 | CR+KLM |
| R110 | | | - | 1068.0 | 3 | 1068.31 | 1068.0 | CR+KLM |
| R111 | | | - | 1045.9 | 39 | 83931.48 | 1048.7 | CR+KLM |
| R112 | | | - | 946.7 | 9 | 202803.94 | 948.6 | **JPSP** |
| C101 | 827.3 | 1 | 3.02 | 827.3 | 1 | 3.02 | 827.3 | KDMSS |
| C102 | 827.3 | 1 | 12.92 | 827.3 | 1 | 12.92 | 827.3 | KDMSS |
| C103 | 826.3 | 1 | 33.89 | 826.3 | 1 | 33.89 | 826.3 | KDMSS |
| C104 | 822.9 | 1 | 4113.09 | 822.9 | 1 | 4113.09 | 822.9 | KDMSS |
| C105 | 827.3 | 1 | 5.34 | 827.3 | 1 | 5.34 | 827.3 | KDMSS |
| C106 | 827.3 | 1 | 7.15 | 827.3 | 1 | 7.15 | 827.3 | KDMSS |
| C107 | 827.3 | 1 | 6.55 | 827.3 | 1 | 6.55 | 827.3 | KDMSS |
| C108 | 827.3 | 1 | 14.46 | 827.3 | 1 | 14.46 | 827.3 | KDMSS |
| C109 | 827.3 | 1 | 20.53 | 827.3 | 1 | 20.53 | 827.3 | KDMSS |
| RC101 | 1584.1 | 59 | 56.62 | 1619.8 | 1 | 12.39 | 1619.8 | KDMSS |
| RC102 | | | - | 1457.4 | 1 | 76.69 | 1457.4 | CR+KLM |
| RC103 | | | - | 1257.7 | 3 | 2705.78 | 1258.0 | CR+KLM |
| RC104 | | | - | 1129.9 | 7 | 65806.79 | 1132.3 | IV |
| RC105 | 1472.0 | 191 | 309.83 | 1513.7 | 1 | 26.73 | 1513.7 | KDMSS |
| RC106 | | | - | 1367.3 | 37 | 15891.55 | 1372.7 | S |
| RC107 | | | - | 1207.8 | 1 | 153.80 | 1207.8 | IV |
| RC108 | | | - | 1114.2 | 1 | 3365.00 | 1114.2 | IV |
| R201 | | | - | 1143.2 | 1 | 139.03 | 1143.2 | KLM |
| R202 | | | - | 1027.3 | 13 | 8282.38 | 1029.6 | **JPSP** |
| R203 | | | - | 870.8 | 1 | 54187.40 | 870.8 | **JPSP** |
| R204 | | | - | | | - | - | - |
| R205 | | | - | | | - | - | - |
| R206 | | | - | | | - | - | - |
| R207 | | | - | | | - | - | - |
| R208 | | | - | | | - | - | - |
| R209 | | | - | 854.8 | 3 | 78560.47 | 854.8 | **JPSP** |
| R210 | | | - | | | - | - | - |
| R211 | | | - | | | - | - | - |
| C201 | 589.1 | 1 | 203.34 | 589.1 | 1 | 203.34 | 589.1 | CR+KLM |
| C202 | 589.1 | 1 | 3483.15 | 589.1 | 1 | 3483.15 | 589.1 | CR+KLM |
| C203 | 588.7 | 1 | 13070.71 | 588.7 | 1 | 13070.71 | 588.7 | KLM |
| C204 | | | - | | | - | 588.1 | IV |
| C205 | 586.4 | 1 | 416.56 | 586.4 | 1 | 416.56 | 586.4 | CR+KLM |
| C206 | 586.0 | 1 | 594.92 | 586.0 | 1 | 594.92 | 586.0 | CR+KLM |
| C207 | 585.8 | 1 | 1240.97 | 585.8 | 1 | 1240.97 | 585.8 | CR+KLM |
| C208 | 585.8 | 1 | 555.27 | 585.8 | 1 | 555.27 | 585.8 | KLM |
| RC201 | | | - | 1261.7 | 3 | 229.27 | 1261.8 | KLM |
| RC202 | | | - | 1092.3 | 1 | 312.57 | 1092.3 | IV+C |
| RC203 | 922.6 | 11 | 34063.95 | 923.7 | 1 | 14917.36 | 923.7 | **JPSP** |
| RC204 | | | - | | | - | - | - |
| RC205 | | | - | 1154.0 | 1 | 221.24 | 1154.0 | IV+C |
| RC206 | | | - | 1051.1 | 1 | 339.69 | 1051.1 | **JPSP** |
| RC207 | | | - | | | - | - | - |
| RC208 | | | - | | | - | - | - |



Table A.8: Instances with 100 customers.

## A.2 Solutions of Closed Solomon Instances

| Cost | Route |
|---|---|
| 8.8 | 53 |
| 119.2 | 70, 30, 20, 66, 65, 71, 35, 34, 78, 77, 28 |
| 105.4 | 92, 98, 91, 44, 14, 38, 86, 16, 61, 85,100,37 |
| 84.1 | 2, 57, 15, 43, 42, 87, 97, 95, 94, 13, 58 |
| 106.5 | 73, 22, 41, 23, 67, 39, 56, 75, 74, 72, 21,4 |
| 114.6 | 52, 88, 62, 19, 11, 64, 63, 90, 32, 10, 31 |
| 78.4 | 6, 96, 59, 99, 93, 5, 84, 17, 45, 83, 60, 8 |
| 107.3 | 26, 12, 80, 68, 29, 24, 55, 4, 25, 54 |
| 93.2 | 27, 69, 76, 3, 79, 9, 51, 81, 33, 50, 1 |
| 114.6 | 18, 7, 82, 8, 46, 36, 49, 47, 48 |
| 932.1 | 10 |

Table A.9: Solution of R108.100. The left column is the cost of the routes and the total cost. The right column is a comma separated list indicating the customers visited on the routes in the order of visit and the total number of routes.

| Cost | Route |
|---|---|
| 815.8 | 2, 41, 22, 75, 56, 23, 67, 39, 25, 55, 54 |
| 101.7 | 28, 76, 79, 78, 34, 35, 71, 65, 66, 20, 1 |
| 128.2 | 31, 62, 19, 11, 63, 64, 49, 36, 47, 48 |
| 62.8 | 53, 40, 21, 73, 74, 72, 4, 26 |
| 198.0 | 52, 88, 7, 82, 8, 46, 45, 17, 84, 5, 89 |
| 96.4 | 12, 80, 68, 24, 29, 3, 77, 50 |
| 100.5 | 61, 16, 86, 38, 14, 44, 91,100, 37, 59, 96 |
| 67.6 | 18, 83, 60, 99, 93, 85, 98, 92, 97, 13 |
| 103.8 | 27, 69, 33, 81, 9, 51, 30, 32, 90, 10, 70 |
| 948.6 | 10 |

Table A.10: Solution of R112.100.

| Cost | Route |
|---|---|
| 8.8 | 53 |
| 93.6 | 52, 62, 63, 90, 10, 32, 70 |
| 177.2 | 83, 45, 82, 48, 47, 36, 19, 11, 64, 49, 46, 17, 5, 60, 89 |
| 223.8 | 50, 33, 65, 71, 29, 76, 3, 79, 78, 81, 9, 51, 20, 66, 35, 34, 68, 77 |
| 140.2 | 27, 69, 1, 30, 31, 88, 7, 18, 8, 84, 86, 91,100, 37, 98, 93, 59, 94 |
| 67.1 | 40, 73, 41, 22, 74, 2, 58 |
| 148.9 | 28, 26, 21, 72, 75, 39, 67, 23, 56, 4, 54, 55, 25, 24, 80, 12 |
| 170.0 | 95, 92, 42, 15, 14, 38, 44, 16, 61, 85, 99, 96, 6, 87, 57, 43, 97, 13 |
| 1029.6 | 8 |

Table A.11: Solution of R202.100.

| Cost | Route |
|---|---|
| 24.2 | 53, 40, 58 |
| 142.1 | 27, 69, 1, 76, 3, 79, 78, 81, 9, 66, 71, 35, 34, 29, 68, 77, 28 |
| 187.3 | 89, 18, 45, 46, 36, 47, 48, 19, 11, 62, 88, 7, 82, 8, 83, 60, 5, 84, 17, 61, 91,100, 37, 98, 93, 59, 94 |
| 183.3 | 95, 92, 97, 42, 15, 43, 14, 44, 38, 86, 16, 85, 99, 96, 6, 87, 57, 41, 22, 74, 73, 2, 13 |
| 190.3 | 50, 33, 51, 71, 65, 20, 30, 32, 90, 63, 64, 49, 10, 70, 31, 52 |
| 143.6 | 26, 21, 72, 75, 39, 67, 23, 56, 4, 55, 25, 54, 24, 80, 12 |
| 870.8 | 6 |

Table A.12: Solution of R203.100.

| Cost | Route |
|---|---|
| 202.5 | 27, 31, 7, 48, 47, 36, 46, 45, 8, 18, 6, 37, 44, 14, 38, 16, 17, 5, 13 |
| 130.5 | 2, 42, 43, 15, 23, 39, 22, 41, 21, 40 |
| 242.5 | 28, 12, 3, 33, 50, 1, 30, 11, 49, 19, 10, 32, 20, 9, 35, 34, 29, 24, 25, 4, 26 |
| 575.5 | 3 |

Table A.13: Solution of R207.50.

| Cost | Route |
|---|---|
| 146.8 | 52, 7, 82, 83, 18, 6, 94, 13, 87, 57, 15, 43, 42, 97, 92, 37,100, 91, 93, 96 |
| 198.7 | 95, 99, 59, 98, 85, 5, 84, 61, 16, 44, 14, 38, 86, 17, 45, 8, 46, 36, 49, 48, 60, 89 |
| 205.9 | 27, 69, 31, 88, 62, 47, 19, 11, 64, 63, 90, 30, 51, 71, 9, 81, 33, 79, 3, 77, 68, 80, 24, 54, 26 |
| 157.6 | 28, 12, 76, 29, 78, 34, 35, 65, 66, 20, 32, 10, 70, 1, 50 |
| 145.8 | 40, 2, 73, 21, 72, 75, 23, 67, 39, 25, 55, 4, 56, 74, 22, 41, 58, 53 |
| 854.8 | 5 |

Table A.14: Solution of R209.100.

| Cost | Route |
|---|---|
| 139.4 | 81, 54, 72, 37, 36, 39, 42, 44, 41, 38, 40, 35, 43, 61, 68 |
| 172.8 | 90, 65, 83, 64, 85, 63, 89, 76, 23, 21, 48, 18, 19, 49, 22, 20, 51, 84, 56, 66 |
| 241.4 | 69, 98, 88, 53, 82, 99, 52, 86, 87, 9, 10, 47, 17, 13, 74, 59, 97, 75, 58, 77, 25, 24, 57 |
| 211.0 | 1, 3, 5, 45, 60, 12, 11, 15, 16, 14, 78, 73, 79, 7, 6, 8, 46, 4, 2, 55,100, 70 |
| 159.1 | 91, 92, 95, 62, 33, 32, 30, 27, 26, 28, 29, 31, 34, 50, 67, 94, 93, 71, 96, 80 |
| 923.7 | 5 |

Table A.15: Solution of RC203.100.

| Cost | Route |
|---|---|
| 8.4 | 90 |
| 186.6 | 81, 94, 67, 84, 85, 51, 76, 89, 48, 25, 77, 58, 74 |
| 168.6 | 92, 71, 72, 42, 39, 38, 36, 40, 44, 43, 41, 37, 35, 54, 93, 96 |
| 180.9 | 65, 83, 64, 95, 62, 63, 33, 30, 31, 29, 27, 28, 26, 32, 34, 50, 56, 91, 80 |
| 189.6 | 61, 2, 45, 5, 8, 7, 79, 73, 78, 53, 88, 6, 46, 4, 3, 1,100, 70, 68 |
| 120.9 | 82, 99, 52, 86, 57, 23, 21, 18, 19, 49, 20, 22, 24, 66 |
| 196.1 | 69, 98, 12, 14, 47, 16, 15, 11, 59, 75, 97, 87, 9, 13, 10, 17, 60, 55 |
| 1051.1 | 7 |

Table A.16: Solution of RC206.100.

# Bibliography


[1] COIN — COmputational INfrastructure for Operations Research, 2007. www.coin-or.org.

[2] V. Chvátal. Edmonds polytopes and hierarchy of combinatorial problems. *Discrete Mathematics*, 4:305–337, 1973.

[3] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.

[4] J.E. Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19:379–394, 1989.

[5] N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Oper. Res. Lett.*, 34(1):58–68, 2006.

[6] A. Caprara, M. Fischetti, and A. N. Letchford. On the separation of maximally violated mod-$k$ cuts. *Mathematical Programming*, 87(A): 37–56, 1999.

[7] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 23(10): 2972–2990, 2005.

[8] W. Cook and J. L. Rich. A parallel cutting plane algorithm for the vehicle routing problem with time windows. Technical Report TR99-04, Computational and Applied Mathematics, Rice University, Houston, Texas, USA, 1999.

[9] E. Danna and C. Le Pape. Accelerating branch-and-price with local search: A case study on the vehicle routing problem with time windows. In G. Desaulniers, J. Desrosiers, , and M. M. Solomon, editors, *Column Generation*, chapter 3, pages 30–130. Springer, 2005.

[10] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

[11] G. Desaulniers, J. Desrosiers, J. Ioachim, I. M. Solomon, F. Soumis, and D. Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T. G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, chapter 3, pages 57–93. Springer, 1998.

[12] M. Desrochers. *La fabrication d'horaires de travail pour les conducteurs d'autobus par une méthode de génération de colonnes*. PhD thesis, Université de Montréal, Canada, 1986.

[13] M. Dror. Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, 42:977–979, 1994.

[14] I. Dumitrescu. *Constrained Path and Cycle Problems*. PhD thesis, Department of Mathematics and Statistics, University of Melbourne, Australia, 2002.

[15] F. Eisenbrand. Note on the membership problem for the elementary closure of a polyhedron. *Combinatorica*, 19(2):297–300, 1999.

[16] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

[17] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R.F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming*, 106 (3):491–511, 2006.

[18] S. Irnich. Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectrum*, 30(1):113–148, January 2008.

[19] S. Irnich and G. Desaulniers. *Shortest Path Problems with Resource Constraints*, chapter 2 in Column Generation (eds. G. Desaulniers and Jacques Desrosiers and M.M. Solomon), pages 33–65. GERAD 25th Anniversary Series. Springer, 2005.

[20] S. Irnich and D. Villeneuve. The shortest-path problem with resource constraints and k-cycle elimination for k $\geq$ 3. *INFORMS J. on Computing*, 18(3):391–406, 2006.

[21] M. Jepsen, B. Petersen, and S. Spoorendonk. A branch-and-cut-and-price framework for the vrp applied on cvrp and vrptw. Master's thesis, DIKU University of Copenhagen, Denmark, 2005.

[22] B. Kallehauge, J. Larsen, and O.B.G. Madsen. Lagrangean duality and non-differentiable optimization applied on routing with time windows

- experimental results. Technical Report Internal report IMM-REP-2000-8, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 2000.

[23] N. Kohl, J. Desrosiers, O. B. G. Madsen, M. M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999.

[24] J. Larsen. *Parallelization of the vehicle routing problem with time windows*. PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1999.

[25] M. E. Lubbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[26] J. Lysgaard. CVRPSEP: A package of separation routines for the capacitated vehicle routing problem. Available at: url: www.asb.dk/∼lys, 2003.

[27] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle problem. *Mathematical Programming*, 100(A):423–445, 2004.

[28] G. Nemhauser and S. Park. A polyhedral approach to edge coloring. *Operations Research Letters*, 10:315–322, 1991.

[29] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, Inc., 1988.

[30] D. Pisinger and L. B. Reinhardt. Multi-objective non-additive shortest path. 2006.

[31] G. Righini and M. Salani. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006.

[32] M. Salani. *Branch-and-Price Algorithms for Vehicle Routing Problems*. PhD thesis, Universitá Degli Studi Di Milano, Facoltá di Scienza Matematiche, Fisuche e Naturali Dipartimento di Technologie dell'Informazione, Milano, Italy, 2005.

[33] M. M. Solomon. Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research*, 35:234–265, 1987.

[34] SPEC. Standard Performance Evaluation Corporation, 2005. www.spec.org.

[35] P. Toth and D. Vigo. *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 2001.

[36] L. A. Wolsey. *Integer Programming*. John Wiley & Sons, Inc., 1998.

# Chapter 3 A Branch-and-Cut Algorithm for the Symmetric Two-echelon Capacitated Vehicle Routing Problem

Mads Jepsen Simon Spoorendonk [1]
*DTU Management Engineering, Technical University of Denmark*
Stefan Røpke[2]
*DTU Transport, Technical University of Denmark*

May 2011

## Abstract

This paper presents an exact method for solving the symmetric two-echelon capacitated vehicle routing problem, a transportation problem concerned with the distribution of goods from a depot to a set of customers through a set of satellite locations. The presented method is based on an edge flow model that is a relaxation and provides a valid lower bound. A specialized branching scheme is employed to obtain feasible solutions. Out of a test set of 93 instances the algorithm is able to solve 47 to optimality surpassing previous exact algorithms.

## 3.1 Introduction

The two-echelon capacitated vehicle routing problem (2E-CVRP) is a distribution and location problem that can be described as: given a depot, a set of satellites, and a set of customers each with a demand, the 2E-CVRP consist in distributing goods from the depot to the customers through the set of satellites. Two vehicle types are considered, a large capacity vehicle type (first echelon vehicles) going between the depot and the satellites and a small capacity vehicle type (second echelon vehicles), based at the satellites, serve the customers. Each customer must be visited by exactly one small

*Supported by the Danish Council for Independent Research | Technology and Production Sciences (project 274-08-0353)
†Supported by the Danish Council for Independent Research | Natural Sciences (project 272-08-0508)

vehicle. Additionally it is feasible to leave one or more satellites unused and each satellite may be visited by several vehicles from the depot in order to meet the demand. No vehicle may be loaded such that the vehicle capacity is exceeded. The number of vehicles to be used of each type is bounded from above and the number of smaller vehicles used at each satellite is also bounded from above. A satellite-specific handling cost is incurred for each unit of freight transported through the satellite. The objective is to minimize the sum of the vehicle travel costs and the handling costs at the satellites.

The 2E-CVRP is relevant in city-logistic applications. Due to legal restrictions it may not be feasible to use large trucks within the center of large cities. Therefore, to distribute goods efficiently it is convenient to use a two-tier distribution network as in the 2E-CVRP where satellite facilities are located at the outskirts of the city. Figure 3.1 shows an example of a solution to a 2E-CVRP instance with three satellites and six customers. The customers are represented by circles, the satellites by triangles and the depot by a square. First and second echelon vehicle routes are represented by bold lines and thin lines, respectively, and the capacity of the first and second echelon vehicles are 100 and 75, respectively. The demand of customers and satellite handling costs are found in tables (a) and (b) of Figure 3.1. The figure illustrates several features of the 2E-CVRP. For example, the delivery to satellite 3 is split between two vehicles and satellite 2 is left unused because of its high handling costs even though it has a favorable position.

The literature on the 2E-CVRP is limited but several contributions have emerged in recent years. Feliu et al. [16] and Perboli et al. [35] present a commodity flow formulation for the 2E-CVRP and solve this model through a branch-and-cut algorithm. The latter paper extend the model presented in the former paper by introducing a maximum limit on the number of vehicles available at each satellite. Perboli et al. [33, 34] present several valid inequalities for the 2E-CVRP. Perboli et al. [35] also present two math-based heuristics derived from the exact approach. Other heuristic methods are presented in Crainic et al. [12, 13, 14]. The latter paper presents a multi-start heuristic which outperforms the previous heuristics.

If the assignment of customers to satellites is given then the 2E-CVRP decomposes into two separate routing problems. In the first echelon we are faced with a split delivery capacitated vehicle routing problem (SDCVRP) while in the second echelon we must solve a capacitated vehicle routing problem (CVRP) for each satellite that is in use. Both routing problems have been studied extensively in the literature. The current state of the art for the CVRP are surveyed by Laporte [22, 23]. In particular, the best exact algorithms for the CVRP are presented by Fukasawa et al. [17], Baldacci et al. [4, 7], Lysgaard et al. [28]. Except the latter which is a branch-and-cut algorithm, the solution approaches are based on decomposition algorithms enhanced with cutting planes. The SDCVRP is surveyed by Archetti and

Speranza [1] and recent exact and lower bounding algorithms are presented by Archetti et al. [2], Jin et al. [21], Moreno et al. [30] as well as by Desaulniers [15] that also includes time window constraints. The 2E-CVRP is related to a number of other two-level routing problems. These include the *location-routing problem* (Laporte et al. [25], Nagy and Salhi [31], Contardo et al. [11], Belenguer et al. [8], Baldacci et al. [6]), the *two-echelon location routing problem* (Nguyen et al. [32]), the *truck-and-trailer problem* (Chao [10], Tan et al. [37]) and the *capacitated m-ring-star problem* (Baldacci et al. [3]).

The main contribution of this work is a new mathematical model that is a relaxation for the 2E-CVRP. The model is inspired by the edge flow formulation of the CVRP introduced by Laporte and Nobert [24] and the SDCVRP formulation proposed by Belenguer et al. [9]. The relaxation has fewer variables than the previously proposed formulation but does have several constraint sets of exponential size. The new relaxation provides a lower bound for the 2E-CVRP but does not necessarily provide feasible solutions. Therefore, we devise a feasibility test and a specialized branching scheme to obtain optimal feasible integer solutions. A branch-and-cut algorithm is developed to solve the 2E-CVRP to optimality using the specialized branch rule. Through computational studies we show that the branch-and-cut algorithm based on the new relaxation outperforms previous exact algorithms.

The paper is organized as follows: Section 3.2 presents a mathematical formulation of the 2E-CVRP and a relaxation to calculate a lower bound for the symmetric 2E-CVRP. Section 3.3 presents an exact branch-and-cut algorithm that utilizes the presented relaxation, feasibility test, and a specialized branching scheme to obtain optimal solutions to the 2E-CVRP. Section 3.4 presents the experimental results and Section 3.5 concludes the paper.

## 3.2 A mathematical formulation and a relaxation

Perboli et al. [35] propose a commodity flow formulation to solve the 2E-CVRP. We start by giving a similar directed three index formulation and will then derive a relaxation from this. The three index formulation differs from that of Perboli et al. [35] as it appears that the model presented in [35] may not provide correct upper bounds when there are more than 2 satellites in the solution. An example of an infeasible integer solution to the model in [35] can be found in Figure 3.2a.

A graph $G = (V_0 \cup V_S \cup V_C, A \cup A')$ is given where the set of nodes $V = V_0 \cup V_S \cup V_C$ is split into three subsets: a set containing the depot ($V_0 = \{0\}$), a set of satellites nodes ($V_S$) and a set of customers ($V_C$). The set of arcs is composed of two subsets $A = A(V_0 \cup V_S)$ and $A' = A(V_S \cup V_C)$ where $A(S), S \subset V$ is the set of all arcs with both endpoints in $S$. The size

of the two fleets associated with the echelons are given as $K$ and $K'$ with capacity $C$ and $C'$, respectively and the set of first echelon vehicles is given as $\bar{K}$. $K'_s$ is the maximum number of second echelon vehicles serviced at satellite $s \in V_S$. Let $d_i$ be the demand of customer $i \in V_C$. For shorthand notation we use $\delta^-(s)$ and $\delta^+(s)$ to denote respectively the entering and leaving first echelon arcs of satellite $s \in V_S$. Similar $\delta'^-(i)$ and $\delta'^+(i)$ are the entering and leaving second echelon arcs of node $i \in V_S \cup V_C$. The constant $M$ is a sufficiently large constant which can be set to $|V_S| + 1$, $h_s$ is the unit handling cost of freight in satellite $s \in V_S$, $c_{ij}$ and $c'_{ij}$ is the travel cost of respectively arcs $(i,j) \in A$ and arcs $(i,j) \in A'$. We assume, throughout this paper, that the travel costs are symmetric and that they satisfy the triangle inequality.

The variables for the first echelon routing problem are defined as follows: $\bar{x}_{ijk} \in \{0,1\}$ is equal to 1 iff vehicle $k$ goes from $i \in V_S$ to $j \in V_S$, $\bar{w}_{sk} \in \mathbb{R}_+$ is the amount of freight delivered to satellite $s$ by vehicle $k$, $\bar{u}_{sk} \in \mathbb{R}_+$ is the "position" of satellite $s \in V_s$ in route $k$. For the second echelon the variables are defined as follows: $\bar{f}_{ijs} \in \mathbb{R}_+$ is the load on a vehicle from satellite $s$ when leaving node $i$, $\bar{z}_{ijs} \in \{0,1\}$ is 1 iff second echelon vehicle from satellite $s$ goes from $i$ to $j$. Finally the variables $\bar{t}_s \in \mathbb{R}_+$ is the total demand delivered from satellite $s$. A three-index formulation for the directed 2E-CVRP is

$$\min \sum_{k \in \bar{K}} \sum_{(i,j) \in A} c_{ij} \bar{x}_{ijk} + \sum_{s \in V_S} \sum_{(i,j) \in A'} c'_{ij} \bar{z}_{ijs} + \sum_{s \in V_S} h_s \bar{t}_s \tag{3.1}$$

$$\text{s.t.} \sum_{(i,j) \in \delta^+(s)} \bar{x}_{ijk} = \sum_{(i,j) \in \delta^-(s)} \bar{x}_{ijk} \qquad \forall s \in V_s, k \in \bar{K} \tag{3.2}$$

$$\sum_{(i,j) \in \delta^+(s)} \bar{x}_{ijk} \leq 1 \qquad \forall s \in V_0 \cup V_s, k \in \bar{K} \tag{3.3}$$

$$\bar{u}_{ik} + 1 \leq \bar{u}_{jk} + M(1 - \bar{x}_{ijk}) \qquad \forall (i,j) \in A(V_S), k \in \bar{K} \tag{3.4}$$

$$\bar{w}_{sk} \leq C \sum_{(i,j) \in \delta^+(s)} \bar{x}_{ijk} \qquad \forall s \in V_s, k \in \bar{K} \tag{3.5}$$

$$\sum_{s \in V_S} \bar{w}_{sk} \leq C \qquad \forall k \in \bar{K} \tag{3.6}$$

$$\sum_{k \in K} \bar{w}_{sk} = \bar{t}_s \qquad \forall s \in V_s \tag{3.7}$$

$$\sum_{s \in V_S} \sum_{(a,b) \in \delta'^+(i)} \bar{z}_{abs} = 1 \qquad \forall i \in V_C \tag{3.8}$$

$$\sum_{(a,b) \in \delta'^-(i)} \bar{z}_{abs} = \sum_{(a,b) \in \delta'^+(i)} \bar{z}_{abs} \qquad \forall i \in V_C, s \in V_S \tag{3.9}$$

$$\sum_{s' \in V_S \setminus \{s\}} \Big( \sum_{(a,b) \in \delta'^{+}(s)} \bar{z}_{abs'} + \sum_{(a,b) \in \delta'^{-}(s)} \bar{z}_{abs'} \Big) = 0 \qquad \forall s \in V_S \tag{3.10}$$

$$\sum_{(a,b) \in \delta'^{+}(s)} \bar{z}_{abs} \leq K'_s \qquad \forall s \in V_S \tag{3.11}$$

$$\sum_{s \in V_S} \sum_{(a,b) \in \delta'^{+}(s)} \bar{z}_{abs} \leq K' \tag{3.12}$$

$$\sum_{s \in V_S} \sum_{(a,b) \in \delta'^{+}(i)} \bar{f}_{abs} = \sum_{s \in V_S} \sum_{(a,b) \in \delta'^{-}(i)} \bar{f}_{abs} + d_i \qquad \forall i \in V_C \tag{3.13}$$

$$\bar{f}_{abs} \leq C' \bar{z}_{abs} \qquad \forall s \in V_S, (a,b) \in A' \tag{3.14}$$

$$\bar{t}_s = \sum_{(a,b) \in \delta'^{-}(s)} \bar{f}_{abs} \qquad \forall s \in V_S \tag{3.15}$$

$$\bar{x}_{ijk} \in \{0,1\} \qquad \forall (i,j) \in A, k \in \bar{K} \tag{3.16}$$

$$\bar{w}_{sk} \geq 0 \qquad \forall s \in V_S, k \in \bar{K} \tag{3.17}$$

$$\bar{u}_{sk} \geq 0 \qquad \forall s \in V_S, k \in \bar{K} \tag{3.18}$$

$$\bar{f}_{abs} \geq 0 \qquad \forall s \in V_S, (a,b) \in A' \tag{3.19}$$

$$\bar{z}_{abs} \in \{0,1\} \qquad \forall s \in V_S, (a,b) \in A' \tag{3.20}$$

$$\bar{t}_s \geq 0 \qquad \forall s \in V_S. \tag{3.21}$$

The objective minimizes the travel cost for the first and second echelon as well as the handling fee in the satellites. Constraints (3.2)-(3.6) are concerned with the routing of the first echelon and corresponds to a compact formulation for the SDCVRP suggested by Jin et al. [20]. Constraints (3.2) ensure flow conservation for each vehicle at each satellite, constraints (3.3) ensure that a vehicle visit a satellite at most once, constraints (3.4) eliminate sub-tours and constraints (3.5) and (3.6) ensure that the capacity of the vehicle is not exceeded. Constraints (3.7) link the delivery from all the vehicles with the delivery variable for each satellite. Constraints (3.8)-(3.15) are concerned with the routing of the second echelon, which is based on the Gavish and Graves [18] one-commodity flow formulation for the multi-depot vehicle routing problem (see [26] for a modern treatment of the one-commodity flow formulation of the CVRP). Constraints (3.8) ensure that a customer is visited, constraints (3.9) ensure conservation of the vehicle origin at each customer, constraints (3.10) eliminate traffic between the satellites, (3.11) and (3.12) limit the number of vehicles out of each satellite and the total amount of vehicles used and constraints (3.13) and (3.14) ensure that the capacity of a second echelon vehicle is not exceeded. Finally constraints (3.15) connect the freight delivered from a satellite to the second echelon with the

freight delivered from the depot to that satellite. Constraints (3.16)-(3.21) specify the domain of the variables.

Since model (3.1)–(3.21) is highly symmetric (see e.g. [29]) because of the $k$ index on the first echelon decision variables and since the LP relaxation of the model tends to provide poor lower bounds we propose an alternative model that is a relaxation of (3.1)–(3.21) but eliminates the symmetry and whose LP relaxation in general provides better lower bounds. In the first echelon we use the relaxation for the SDCVRP suggested by Belenguer et al. [9] and incorporate the freight variables. In the second echelon a modified version of the mathematical model for the capacitated location routing problem introduced by Contardo et al. [11] is used. The two models are linked by ensuring that the load delivered to each satellite matches the load delivered to the customers served from that satellite.

To describe the model we use the same notation as previously defined unless otherwise noted. Let $E$ and $E'$ denote the edge sets for respectively the first and second echelon, and let $c_e$ (respectively $c'_e$) be the cost of traversing an edge $e \in E$ (respectively $e \in E'$). We use $\delta(S) \subset E$ (respectively, $\delta'(S) \subset E'$ ) to denote the edges with exactly one endpoint in $S \subset V_0 \cup V_S$ (respectively, $S \subset V_S \cup V_C$) and we let $E(S_1 : S_2) \subset E$ (respectively, $E'(S_1 : S_2) \subset E'$) denote the set of edges between the two disjoint sets of nodes $S_1, S_2 \subset V_0 \cup V_S$ (respectively, $S_1, S_2 \subset V_S \cup V_C$).

In the following we describe the decision variables used in the model. The variables connected to the first echelon are the integer variables $\alpha_s$ for all $s \in V_S$ which equal the number of visit to the satellite and integer variables $x_e$ for all $e \in E$ which equal the number of times the edge is traversed. The variables $l_s$ equals the amount of freight delivered to satellite $s \in V_S$. In the second echelon the binary variables $\beta_s$ for all $s \in V_S$ indicate if the satellite is visited, the binary variables $y_e$ for all $e \in E'$ indicate if the edge is traversed and finally the binary variables $z_e$ for all $e \in E(V_S : V_C)$ indicate if an edge between a satellite and a customer is used twice (in case of routes containing only one customer). We note that for an edge $e \in E(V_S : V_C)$ the two variables $y_e$ and $z_e$ cannot both be one simultaneously. For $S \subseteq V_C$ we define $r(S) = \left\lceil \sum_{i \in S} d_i / C' \right\rceil$, that is, $r(S)$ is a lower bound on the number of vehicles needed to service the customers in $S$. With these definitions we propose the following relaxation R for the 2E-CVRP:

$$\min \sum_{e\in E} c_e x_e + \sum_{e\in E'} c'_e y_e + \sum_{e\in\delta'(V_S)} 2c'_e z_e + \sum_{s\in V_S} h_s l_s \tag{3.22}$$

$$\text{s.t.} \sum_{e\in\delta(s)} x_e = 2\alpha_s \qquad \forall s \in V_S \tag{3.23}$$

$$\sum_{e\in\delta(V_0)} x_e \leq 2K \tag{3.24}$$

$$\sum_{s\in V_S} l_s = \sum_{i\in V_C} d_i \tag{3.25}$$

$$\beta_s \leq \alpha_s \qquad \forall s \in V_S \tag{3.26}$$

$$\sum_{e\in\delta(S)} x_e \geq \frac{2}{C} \sum_{i\in S} l_i \qquad \forall S \subseteq V_S \tag{3.27}$$

$$\sum_{e\in\delta'(i)} y_e + \sum_{e\in E'(i:V_S)} 2z_e = 2 \qquad \forall i \in V_C \tag{3.28}$$

$$\sum_{e\in\delta'(V_S)} (y_e + 2z_e) \leq 2K' \tag{3.29}$$

$$\sum_{e\in\delta'(i)} (y_e + 2z_e) \leq 2K'_s\beta_s \qquad \forall s \in V_S \tag{3.30}$$

$$y_e + z_e \leq 1 \qquad \forall e \in \delta(V_S) \tag{3.31}$$

$$\sum_{e\in\delta'(S)} y_e + \sum_{e\in E'(S:V_S)} 2z_e \geq 2r(S) \qquad \forall S \subseteq V_C, |S| \geq 2 \tag{3.32}$$

$$\sum_{e\in\delta'(S)} y_e \geq \sum_{e\in E(\{i\}:I)} 2y_e + \sum_{e\in E(\{j\}:V_s\setminus I)} 2y_e \quad \forall S \subseteq V_C, |S| \geq 2,$$

$$\forall i, j \in S, \forall I \subset V_S, \tag{3.33}$$

$$\sum_{e\in E(V_s:\{i\})} y_e \leq 1 \qquad \forall i \in V_c \tag{3.34}$$

$$\sum_{e\in\left\{\substack{E'(S:(V_C\setminus S)\\ \cup(V_S\setminus\{j\}))}\right\}} y_e + \sum_{e\in E'(S:V_S\setminus\{j\})} 2z_e \geq \frac{2}{C'}\left(\sum_{i\in S} d_i - l_j\right) \quad \forall S \subseteq V_C, \forall j \in V_S$$

$$\tag{3.35}$$

$$x_e \in \mathbb{Z}^+ \qquad \forall e \in E \tag{3.36}$$

$$y_e \in \{0,1\} \qquad \forall e \in E' \tag{3.37}$$

$$z_e \in \{0,1\} \qquad \forall e \in \delta'(V_S) \tag{3.38}$$

$$\alpha_i \in \mathbb{Z}^+ \qquad \forall i \in V_S \tag{3.39}$$

$$\beta_i \in \{0,1\} \qquad \forall i \in V_S \tag{3.40}$$

$$l_s \geq 0 \qquad \forall s \in V_S. \tag{3.41}$$

The objective function (3.22) minimizes travel cost of both the first and the second echelon vehicles as well as the handling costs in the satellites.

The routing in the first echelon is described by constraints (3.23)-(3.27) and the variable domains (3.36) and (3.39)-(3.41). The formulation of the first echelon is inspired by the edge flow formulation for the SDCVRP by Belenguer et al. [9] where the satellites in the 2E-CVRP correspond to the customers in the SDCVRP and the freight delivered are variables instead of constant demand. Constraints (3.23) connects the satellite usage variable $\alpha_s$ to the edge variables $x_e$ corresponding to edges adjacent to satellite $s$, constraint (3.24) ensures that at most $K$ vehicles leaves the depot thereby enforcing that the number of first echelon vehicles is not exceeded and constraint (3.25) ensures that the quantity delivered for distribution is equal to customer demands. Constraints (3.26) ensure that a satellite cannot be used as a distribution point for the second echelon vehicles unless it is visited by at least on vehicle in the first echelon. A visit to a satellite implies (due to the minimization of travel cost) that a positive amount of freight has been delivered to that satellite. Constraints (3.27) are adapted from the one-vehicle generalized large multi-star (GLM) inequalities for the CVRP, see e.g. Letchford and Salazar-González [26]. The constraints impose that the number of vehicles and their capacity visiting a set of satellites is sufficient compared to the amount of freight delivered.

The routing in the second echelon is defined by the constraints (3.28)-(3.35) and the variable domains (3.37)-(3.38). Constraints (3.28) ensure that all customers are visited exactly once, constraints (3.29) and (3.30) ensure that at most $K'$ vehicles are used to service the customers and at most $K'_s$ vehicles are used from each satellite $s$, and constraints (3.31) ensure that an edge can only be used twice if it is a one-customer route. Constraints (3.32) are adapted from the capacity inequalities for the CVRP to ensure that the capacity of the vehicles is not exceeded and that no sub-tours exists between the customers. However, the constraints are not enough to enforce that each second echelon vehicle returns to the satellite it started from. Constraints (3.33)-(3.34) ensure that there cannot exist a solution where there is a path between two satellites, see Figure 3.3 for and example. Constraints (3.33) correspond to the *path elimination constraints* of Belenguer et al. [8] and were introduced in a symmetric version by Contardo et al. [11] for the location routing problem. Since constraints (3.33) are only valid for a subset of two or more customers, constraints (3.34) are added to ensure that a single customer is not connected to more than one satellite. It remains to be ensured that enough freight has been delivered to each satellite such that the demand of the customers serviced by the satellite can be met. This is done by constraints (3.35) that are also adapted from the one-vehicle GLM inequalities. If there is not sufficient freight in the satellite to meet the demand of a set of customers then additional vehicle visits to the customer set is enforced (the right-hand side becomes positive).

The constraints of the first echelon only provides a relaxation of the first echelon routing problem. In an integer solution to (3.22)–(3.41) it can happen that the selected first echelon edges cannot be mapped to a set of feasible vehicle routes. Figure 3.2b depicts an integer solution to constraints (3.23)-(3.27) which is not a feasible routing of the first echelon vehicles. One see, that the four satellites with demand 6 and 7 have to be served by separate vehicles due to the capacity constraint and the fact that split deliveries are not possible for these satellites (only two edges are adjacent to each of the nodes). It is therefore impossible to construct a three-vehicle SDCVRP solution that corresponds to the solution shown on the figure.

For the SDCVRP Belenguer et al. [9] suggest to check the feasibility of an integer solution using a three-index formulation. We adapt a similar approach, see Section 3.3.2 for details.

To summarize, a feasible, optimal solution to R is not necessarily a feasible solution to 2E-CVRP because of the modeling of the first echelon routing decisions. In the following we will show how the model R can be used to obtain a feasible and optimal solution to 2E-CVRP by using special branching rules when infeasible 2E-CVRP solutions are encountered.

## 3.3 A branch-and-cut algorithm

We devise a branch-and-cut algorithm based on R to solve the symmetric 2E-CVRP to optimality. First, R is relaxed by removing the exponential number of constraints (3.27), (3.32), (3.33), and (3.35). The constraints are added dynamically when violated. Since R only provides a lower bound to the 2E-CVRP all integer solutions encountered during the branch-and-cut algorithm must be tested for feasibility. If such an integer solution is in fact infeasible a specialized branching scheme is invoked. Next, we will in detail describe the separation procedures and their complexity, how to verify if an integer solution to R is feasible for the 2E-CVRP, and how to branch on an infeasible integer solution to R.

### 3.3.1 Separation results

In the following, let $(x^*, y^*, z^*, \alpha^*, \beta^*, l^*)$ be an optimal LP solution to the relaxed problem R. The constraints (3.27) are derived from the one-vehicle GLM inequalities for the CVRP. The one-vehicle GLM inequalities can be separated in polynomial time (see Letchford and Salazar-González [26]) and not surprisingly we obtain a similar result for (3.27).

**Theorem 1.** Separating the most violated constraint (3.27) corresponds to

finding $S \subseteq V_S$ such that

$$\sum_{e \in \delta(S)} x_e^* - \frac{2}{C} \sum_{i \in S} l_i^*$$

is minimized. The separation problem is polynomially solvable.

*Proof.* Let $M = \min\{\sum_{i \in V_C} d_i, CK\}$ be the upper bound on the amount of freight that can be delivered at a satellite. Multiply by $C$ and add $2\sum_{i \in V_S} M$ on both sides of the inequality (3.27)? to obtain

$$C \sum_{e \in \delta(S)} x_e^* + 2 \sum_{i \in V_S} M < 2 \sum_{i \in S} l_i^* + 2 \sum_{i \in V_S} M \quad \Leftrightarrow$$
$$C \sum_{e \in \delta(S)} x_e^* + 2 \sum_{i \in S} (M - l_i^*) + 2 \sum_{i \in V_S \setminus S} M < 2 \sum_{i \in V_S} M.$$

Since the depot can never be part of $S$ the problem can be solved as a series of minimum $st$-cut problems on a weighted graph $G(V^*, E^*)$, where $V^* = V_0 \cup V_S^*$ , $V_S^* = \{i \in V_S \mid l_i^* > 0\}$, $E^* = \{e \in E(V^*) \mid x_e^* > 0\}$, $s \in V_S$ and $t \in V_0$. The edge weights are given as

$$w_e = \begin{cases} Cx_e^* & e \in E^*(V^* \setminus \{s,t\}) \\ Cx_e^* + 2(M) & e \in E(s : V_S^* \setminus \{s\}) \\ Cx_e^* + 2(M - l_i^*) & e \in E(t : V_S^*) \end{cases}.$$

A minimization over all satellites results in at most $|V_S|$ minimum $st$-cut problems. □

An example of the graph used for separation of constraints (3.27) is show in Figure 3.4b.

The constraints (3.32) are derived directly from the capacity inequalities known from the CVRP and we obtain our separation results from there.

**Theorem 2.** Separating the most violated constraint (3.32) corresponds to finding $S \subseteq V_C, |S| \geq 2$ such that

$$\sum_{e \in \delta'(S)} y_e^* + \sum_{e \in E'(S:V_S)} 2z_e^* - 2r(S)$$

is minimized. The separation problem is $\mathcal{NP}$-hard when $r(S) = \left\lceil \sum_{i \in S} d_i / C' \right\rceil$.

*Proof.* By shrinking all satellites and the depot into a single super-node an equivalent graph can be constructed that consists of one super-node (a depot) and the customer nodes. The edge weights on edges connected to the super-node is also aggregated such that it equals the combined weight of from edges from a node to the satellites (for each node). Separating a violated capacity inequalities for the CVRP on the constructed graph is identical to solving the separation problem above. This is known to be $\mathcal{NP}$-hard (see e.g. Lysgaard et al. [28]). □

**Theorem 3.** Separating the most violated constraint (3.33) corresponds to finding $S \subseteq V_C$, $|S| \geq 2$,$i, j \in V_C$, $I \subset V_S, i \neq j$, such that

$$\sum_{e \in \delta'(S)} y_e - \sum_{e \in E(\{i\}:I)} 2y_e + \sum_{e \in E(\{j\}:V_s \setminus I)} 2y_e$$

is minimized. The separation problem is polynomially solvable.

*Proof.* The separation is done by considering all pairs of nodes $i, j \in V_C$. Once $i, j \in V_C$ is fixed the set $I \subset V_S$ can be determined in linear time and the set $S \subseteq V_C$ is determined by solving a minimum $st$-cut (see Contardo et al. [11] for details). □

As mentioned earlier, the constraints (3.35) are also similar to the one-vehicle GLM inequalities for the CVRP, so we adapt the results of Letchford and Salazar-González [26].

**Theorem 4.** Separating the most violated constraint (3.35) corresponds to finding $S \subseteq V_C, j \in V_S$ such that

$$\sum_{e \in E'(S:(V_C \setminus S) \cup (V_S \setminus \{j\}))} y_e^* + \sum_{e \in E'(S:V_S \setminus \{j\})} 2z_e^* - \frac{2}{C'} \left( \sum_{i \in S} d_i - l_j^* \right)$$

is minimized. The separation problem is polynomially solvable.

*Proof.* For a given satellite $j \in V_S$, multiply by $C'$ and add $2 \sum_{i \in V_C} d_i$ on both side of the inequality (3.35) to obtain

$$C' \sum_{e \in \left\{ \substack{E'(S:(V_C \setminus S)) \\ \cup (V_S \setminus \{j\}))} \right\}} y_e^* + C' \sum_{e \in E'(S:V_S \setminus \{j\})} 2z_e^* + 2 \sum_{i \in V_C} d_i < 2 \left( \sum_{i \in S} d_i - l_j^* \right) + 2 \sum_{i \in V_C} d_i \quad \Leftrightarrow$$

$$C' \sum_{e \in \left\{ \substack{E'(S:(V_C \setminus S)) \\ \cup (V_S \setminus \{j\}))} \right\}} y_e^* + C' \sum_{e \in E'(S:V_S \setminus \{j\})} 2z_e^* + 2 \sum_{i \in V_C \setminus S} d_i < -2l_j^* + 2 \sum_{i \in V_C} d_i$$

Let $t$ be a super node consisting of the merged satellite nodes in the set $V_S^* \setminus \{j\}$ where $V_S^* = \{i \in V_S \setminus \{j\} \mid \alpha_i^* > 0\}$ and let there exist an edge between $t$ and $i \in V_C$ if $\{e \in E'(V_S^* : i) \mid y_e^* > 0\} \neq \emptyset$. Denote this edge set by $E'^*(t : V_C)$. Set the source node $s = j$. Construct a weighted graph $G(V^*, E'^*)$ where $V^* = \{s, t\} \cup V_C$, and $E'^* = E'^*(V_C) \cup E'^*(s : V_C) \cup E'^*(t : V_C)$ with $E'^*(V_C) = \{e \in E'(V_C) \mid y_e^* > 0\}$ and $E'^*(s : V_C) = E'(s : V_C)$. The edge weights are given as

$$w_e = \begin{cases} C' y_e^* & e \in E'^*(V_C) \\ C' \sum_{e \in E'(V_S^* \setminus \{s\}:i)} (y_e^* + 2z_e^*) & i \in V_C \wedge e \in E'^*(t : V_C) \\ d_i & i \in V_C \wedge e \in E'^*(s : V_C) \end{cases}$$

For a given source node, a satellite, the right-hand side is a constant which leads to at most $|V_S|$ minimum $st$-cut problems that needs to be solved. □

An example of the graph used for separation of constraints (3.35) can be seen in Figure 3.4c.

### 3.3.2 Testing feasibility of integer solutions

As discussed earlier, an integer solution to model R is not necessarily a feasible solution to the symmetric 2E-CVRP. Recall the example given in Figure 3.2b. For the SDCVRP Belenguer et al. [9] suggest to check the feasibility of a solution to an edge flow model by mapping it into a three-index formulation. In a similar fashion we test the feasibility of an integer solution $(x^*, y^*, z^*, \alpha^*, \beta^*, l^*)$ for R by mapping the first echelon part $(x^*, l^*)$ of the solution into the first echelon part of the three-index formulation from Section 3.2. Let $S_i$ for all $i \in V_S$ denote the set of customers serviced by satellite $i$ (implying $l_i^* = \sum_{j \in S_i} d_j$ for the freight solution variable $l_i^*$) and let $x_e^*$ be the solution values of the first echelon edges from R. Recall $\bar{x}_{ijk}$ is equal to 1 iff vehicle $k \in K$ uses the arc from $i \in V_S \cup V_0$ to $j \in V_S \cup V_0$, the continuous variable $\bar{w}_{ik}$ equals the amount of freight delivered at satellite $i \in V_S$ by vehicle $k \in K$, and finally the continuous variables $\bar{u}_{ik}$ for all $i \in V_S \cup V_0$ for each $k \in K$ are used to eliminate sub-tours. The feasibility problem FP is

$$\sum_{(i,j)\in\delta^+(s)} \bar{x}_{ijk} = \sum_{(i,j)\in\delta^-(s)} \bar{x}_{ijk} \qquad \forall s \in V_s, k \in \bar{K} \quad (3.42)$$

$$\sum_{(i,j)\in\delta^+(s)} \bar{x}_{ijk} \leq 1 \qquad \forall s \in V_0 \cup V_s, k \in \bar{K} \quad (3.43)$$

$$\bar{u}_{ik} + 1 \leq \bar{u}_{jk} + M(1 - \bar{x}_{ijk}) \qquad \forall (i,j) \in A(V_S), k \in \bar{K} \quad (3.44)$$

$$\bar{w}_{ik} \leq \min\left\{\sum_{j\in S_s} d_j, C\right\} \sum_{(i,j)\in\delta^+(s)} \bar{x}_{ijk} \qquad \forall s \in V_S, \forall k \in K \quad (3.45)$$

$$\sum_{i\in V_S} \bar{w}_{ik} \leq C \qquad \forall k \in K \quad (3.46)$$

$$\sum_{k\in K} \bar{w}_{ik} = \sum_{j\in S_i} d_j \qquad \forall i \in V_S \quad (3.47)$$

$$\sum_{k\in K} (\bar{x}_{jik} + \bar{x}_{ijk}) = x_e^* \qquad \forall e(i,j) \in E \quad (3.48)$$

$$\bar{x}_{ijk} \in \{0,1\} \qquad \begin{smallmatrix} i,j\in V_S\cup V_0, \\ i\neq j, \forall k\in K \end{smallmatrix} \quad (3.49)$$

$$\bar{u}_{ik} \in \mathbb{R}^+ \qquad \forall i \in V_S, \forall k \in K. \quad (3.50)$$

Constraints (3.42)-(3.46) are identical to constraints (3.2)-(3.6) with the slight difference in constraints (3.45) where the delivery to a node is bounded by the vehicle capacity and the actual delivery to the second echelon customers given in the solution tested. Constraints (3.47) ensure that the demand allocated to the satellite is picked up and constraints (3.48) fixes the arcs to the solution of R. The domains of the variables are defined in (3.49) and (3.50).

A feasible solution to FP provides the routes of the first echelon and the solution to R is feasible. If on the other hand FP is infeasible, additional branch decisions must be imposed. The equivalent problem to FP for the SDCVRP is in general $\mathcal{NP}$-hard, see Belenguer et al. [9]. This is also the case for FP but when the number of satellites is small it is easy to solve.

### 3.3.3 Branching on infeasible integer solutions

Let $(x^*, y^*, z^*, \alpha^*, \beta^*, l^*)$ be a feasible solution to R but infeasible for the 2E-CVRP. In this case we split the sub-problem in the current branch-and-bound node into two new sub-problems based on the solution to the second echelon

$$(y = y^* \wedge z = z^*) \quad \vee \quad (y \neq y^* \vee z \neq z^*)\,.$$

In the first branch the second echelon is fixed. This implies that the demand delivered to the satellites is fixed as well ($l = l^*$). Therefore we can find the optimal solution to the 2E-CVRP for the first branch by solving FP without the arc fixing constraints (3.48) and with an appropriate objective function, i.e., the travel cost of the first echelon edges plus the handling cost and the cost of the second echelon edges. The two latter are constants since the solution of the second echelon is fixed.

Let $E'_{y^*} = \{e \in E' \mid y^*_e = 1\}$ and $E'_{z^*} = \{e \in E' \mid z^*_e = 1\}$. To evaluate the second branch we note that if $\sum_{e \in E'_{y^*}} y_e + \sum_{e \in E'_{z^*}} 2z_e = \left|E'_{y^*}\right| + 2\left|E'_{z^*}\right|$ then $(y = y^* \wedge z = z^*)$. As the degree constraints (3.28) in the second echelon solution are saturated no additional edges can be used. This implies that we can enforce $(y \neq y^* \vee z \neq z^*)$ by adding the branching constraint

$$\sum_{e \in E'_{y^*}} y_e + \sum_{e \in E'_{z^*}} 2z_e \leq \left|E'_{y^*}\right| + 2\left|E'_{z^*}\right| - 1$$

to the sub-problem.

### 3.3.4 Valid inequalities based on connections between echelons

The model R can be strengthened by two simple sets of valid inequalities of polynomial size. The inequalities force lower and upper bounds on the number of second echelon vehicles depending on the amount of freight delivered to a satellite. The first set of inequalities imposes an upper bound on the number of vehicles leaving satellite $j$ depending on the freight $l_j$ available.

**Theorem 5.** Inequalities

$$\sum_{e(i,j) \in \delta'(j)} d_i(y_e + z_e) \leq l_j \qquad \forall j \in V_S \tag{3.51}$$

are valid for R.

*Proof.* Let $S \subseteq V_C$ be the customers serviced by satellite $j$. Then $\sum_{e(i,j)\in\delta'(j)} d_i(y_e + z_e) \leq \sum_{i\in S} d_i$. Since the freight delivered in $j$ is an upper bound on the freight deliverable to customers serviced by $j$ the result follows. □

The second set of inequalities imposes a lower bound on the number vehicles leaving satellite $j$ again depending on the freight $l_j$ delivered.

**Theorem 6.** Inequalities

$$\sum_{e\in\delta'(j)} (y_e + 2z_e) \geq \frac{2}{C'} l_j \qquad \forall j \in V_S \tag{3.52}$$

are valid for R.

*Proof.* The right-hand side calculates a fractional amount of second echelon vehicles needed to deliver the freight $l_j$ at satellite $j$ to the customers. The left-hand side calculates the number of second echelon vehicles leaving and entering satellite $j$ which we can then bound from below. □

### 3.3.5 Obtaining an initial solution

To obtain an initial feasible solution we have implemented a simple heuristic for the 2E-CVRP. The heuristic first solves a multi-depot vehicle routing problem (MDVRP), using the 2E-CVRP customers as the MDVRP customers and the 2E-CVRP satellites as the MDVRP depots. The MDVRP solution will define the second echelon solution, and the solution to the first echelon is found by solving an SDCVRP with customers corresponding to satellites in the 2E-CVRP and customer demands defined by the assignment of customers to satellites in the second echelon solution.

The MDVRP is solved using the adaptive large neighborhood search (ALNS) heuristic by Pisinger and Ropke [36]. The ALNS heuristic only handles constraints on the number of vehicles per depot and therefore the MDVRP solution may be infeasible as a solution to the second echelon of the 2E-CVRP. In this case the MDVRP instance is modified such that the sum of vehicles available on all depots equals the global number of vehicles $K$. We do this by finding the $K$ routes with most customers in the MDVRP solution and assign vehicles to the satellites based on where these routes originate. The modified MDVRP solution is resolved and provides a feasible second echelon routing. The SDCVRP instance is solved to optimality using CPLEX and the mathematical formulation from Jin et al. [20].

## 3.4 Computational experiments

The computational experiments have been carried out on a Intel(R) Xeon X5550 2.67GHz with 24 GB of memory and 8 cores. Our algorithm does not

take advantage of the multiple cores but runs in a single thread. The time limit has been set to 10000 seconds including the time to obtain an initial heuristic solution. The branch-and-cut algorithm is implemented using the callback functions in CPLEX 12.1. We separate the capacity cuts for the customers, i.e., constraints (3.32), using the capacity cuts separation procedure for the CVRP provided in the CVRPSEP package by Lysgaard [27]. We also use this package to separate the framed capacity cuts, GLM inequalities, hypo-tours, and strengthened comb inequalities as was also done by Perboli et al. [34]. Constraints (3.32), constraints (3.33), and the CVRP cuts are added when violated by at least 0.1. Constraints (3.27) are added when violated by at least $1/C$, and constraints (3.35) are added when violated by at least $1/C'$. All cuts are added locally in the branch tree. That is, only to the current branch node and its descendants. This is done to reduce the size of the LPs solved in each branch node. Valid inequalities (3.51) and (3.52) are added a priori.

We let CPLEX handle the branching but force the use of strong branching. In case we meet an integer solution to model (3.22)–(3.41) for which the feasibility problem (3.42)–(3.50) is infeasible we store the search tree node with the solution value and wait until the algorithm has finished to determine if the branch node must be evaluated. During the experiments we never encountered any infeasible nodes with a better lower bounds than the best feasible solution.

### 3.4.1 The test instances

The experiments have been run on the instances in data set 2 and 3 introduced by Feliu et al. [16] and the data set 4 by Crainic et al. [13]. The instances have 21, 32 and 50 customers. Set 2 and 3 are generated from the E instances for the CVRP. The first level vehicles have a factor of 2.5 larger capacity than the second level vehicles which have capacity equal to the capacity in the corresponding CVRP instance. The distance matrix is calculated as the Euclidean distance and distances are neither rounded nor truncated. The handling costs are set to zero in all instances. The satellites in set 2 have been chosen randomly while the satellites in set 3 have been chosen on the border of the bounding box for the customers. Set 4 consists of 54 instances which all have 50 customers. Instances 1-18 have 2 satellites, 19-36 have 3 satellites and 37-54 have 5 satellites. The satellites are distributed after a *random*, *sliced* or *forbidden* location principle. In the sliced location principle the customers are divided into $|V_s|$ slices and a random position for a satellite is chosen in each slice. In the forbidden location principle some zones have been forbidden and satellites are placed outside these. Customers are placed *randomly*, randomly in *centroids* or randomly in *quadrants*. In the centroid distribution a fixed number of customers are placed in a number of different squares and in the quadrants distribution

| Instance set | Class | Sats | No. | Perboli et al. [35] | This paper |
|---|---|---|---|---|---|
| Set 2 | E-n22-k4 | 2 | 6 | 6 | 6 |
| | E-n33-k4 | 2 | 6 | 1 | 6 |
| | E-n51-k5 | 2 | 6 | 0 | 3 |
| | E-n51-k5 | 4 | 3 | 0 | 3 |
| | All | | 21 | 7 | 18 |
| Set 3 | E-n22-k4 | 2 | 6 | 6 | 6 |
| | E-n33-k4 | 2 | 6 | 0 | 5 |
| | E-n51-k5 | 2 | 6 | 0 | 3 |
| | All | | 18 | 6 | 14 |
| Set 4 | 1-18 | 2 | 18 | - | 7 |
| | 19-36 | 3 | 18 | - | 6 |
| | 37-54 | 5 | 18 | 0* | 2 |
| | All | | 54 | 0 | 15 |
| All sets | | | 93 | 13 | 47 |

Table 3.1: Summary of instances solved to optimality. *Perboli et al. [35] considered $K'_s = K', s \in V_S$ instead of the values given in the instance files.

customers are clustered in small zones.

### 3.4.2 Solving instances to optimality

Table 3.1 summarizes the number of solved instances in each set compared to the number of instances solved by Perboli et al. [35] and Perboli et al. [34]. In both papers they used an Intel 3GHz Pentium with 1 GB of memory and the XPress 2008 optimization software. Their time limit was 10000 seconds. The table is split in sections for each set (column "Instance set") and divided further for each instance class (column "Class") and the number of satellites in that class (column "Sats"). The last four columns provide the total number (column "No.") of instances considered in each class, the number solved by Perboli et al. [35] and in this paper (column "This paper"). A "-" indicates that the authors did not consider the instance set. Out of 93 instances we are able to solve 47 to optimality within the time limit. 34 of the solved instances are solved to optimality for the first time. Table 3.2 presents a more detailed analysis of which type of instances in set 4 we were able to solve. The table is divided into a grid based on the distribution of the satellites (rows "DSat") and the customers (columns "DCust"). The results indicate that the centroid distribution of customers provide the easiest instances compared to the other two. This is followed by the random and quadrants distribution which seems equally hard to solve. Regarding the distribution of the satellites, the forbidden distribution results in the most solved instances followed by the random and the sliced

| DSat / DCust | Random | Centroids | Quadrants |
|---|---|---|---|
| Random | 1 | 3 | 0 |
| Sliced | 1 | 2 | 1 |
| Forbidden | 2 | 3 | 2 |

Table 3.2: Distribution of instances in set 4 solved to optimality

| Instance | Satellites | Root LB | Best LB | Gap | HUB | UB | HTime | BACTime |
|---|---|---|---|---|---|---|---|---|
| E-n22-k4 | 6,17 | 403.16 | 417.07 | 0.00% | 417.07 | 417.07 | 4.73 | 0.21 |
| | 8,14 | 377.27 | 384.96 | 0.00% | 384.96 | 384.96 | 4.66 | 1.01 |
| | 9,19 | 425.52 | 470.60 | 0.00% | 532.42 | 470.60 | 4.80 | 12.39 |
| | 10,14 | 359.53 | 371.50 | 0.00% | 371.50 | 371.50 | 2.48 | 1.16 |
| | 11,12 | 404.59 | 427.22 | 0.00% | 444.66 | 427.22 | 4.68 | 3.19 |
| | 12,16 | 376.78 | 392.78 | 0.00% | 392.78 | 392.78 | 4.84 | 1.98 |
| E-n33-k4 | 1,9 | 637.75 | 730.16 | 0.00% | 730.16 | 730.16 | 8.50 | 49.42 |
| | 2,13 | 638.60 | 714.63 | 0.00% | 714.63 | 714.63 | 8.35 | 34.17 |
| | 3,17 | 644.10 | 707.48 | 0.00% | 801.17 | 707.48 | 8.96 | 1126.84 |
| | 4,5 | 682.38 | 778.74 | 0.00% | 778.74 | 778.74 | 8.36 | 54.89 |
| | 7,25 | 650.56 | 756.85 | 0.00% | 756.85 | 756.85 | 8.41 | 87.46 |
| | 14,22 | 685.55 | 779.05 | 0.00% | 825.05 | 779.05 | 8.35 | 2.40 |
| E-n51-k5 | 2,17 | 553.33 | 570.45 | 4.74% | 597.49 | 597.49 | 15.69 | - |
| | 4,46 | 514.61 | 530.76 | 0.00% | 543.25 | 530.76 | 15.63 | 13.25 |
| | 6,12 | 523.74 | 545.76 | 1.66% | 554.81 | 554.81 | 15.94 | - |
| | 11,19 | 548.13 | 581.64 | 0.00% | 606.30 | 581.64 | 15.69 | 213.61 |
| | 27,47 | 513.88 | 534.18 | 0.76% | 538.22 | 538.22 | 15.46 | - |
| | 32,37 | 528.88 | 552.28 | 0.00% | 552.28 | 552.28 | 15.55 | 2113.98 |
| E-n51-k5 | 2,4,17,46 | 500.92 | 530.76 | 0.00% | 548.41 | 530.76 | 18.11 | 84.03 |
| | 6,12,32,37 | 503.09 | 531.92 | 0.00% | 546.33 | 531.92 | 18.11 | 3642.79 |
| | 11,19,27,47 | 504.13 | 527.63 | 0.00% | 577.25 | 527.63 | 18.35 | 798.71 |

Table 3.3: Detailed lower and upper bound results for data set 2. A "-" in time indicates the instance was not solved within 10000 CPU seconds.

distributions. Set 4 shows that when the number of satellites grows to five the instances become much harder to solve. Only two instances with five satellites are solved while we are able to solve seven instances with two satellites and eight with three satellites

The detailed results for all the instances can be found in Tables 3.3, 3.4, and 3.5. The tables list the instance name (column "Instance"), and the satellite index (column "Satellites") for set 2 and 3, and for set 4 it lists the number of satellites (column "Sats") and their the distribution patterns for satellites (column "DSat") and customers (column "DCust"). The next columns in the tables are the root lower bound (column "Root LB"), the best lower bound obtained (column "Best LB"), the gap between the lower and upper bound at the end of the optimization (column "Gap"), the initial upper bound found by the heuristic (column "HUB"), and the best upper bound obtained (column "UB"). The two last columns show the running time of the heuristic (column "HTime") and the running time of the branch-and-cut algorithm (column "BACTime"). We solve all instances with 21 customers in less than 65 seconds, for the 32 customers instances the time

| Instance | Satellites | Root LB | Best LB | Gap | HUB | UB | HTime | BACTime |
|---|---|---|---|---|---|---|---|---|
| E-n22-k4 | 13,14 | 516.55 | 526.15 | 0.00% | 537.57 | 526.15 | 4.70 | 3.20 |
| | 13,16 | 511.93 | 521.09 | 0.00% | 526.11 | 521.09 | 4.81 | 2.32 |
| | 13,17 | 463.65 | 496.38 | 0.00% | 496.38 | 496.38 | 4.73 | 1.07 |
| | 14,19 | 454.58 | 498.80 | 0.00% | 523.59 | 498.80 | 4.70 | 61.19 |
| | 17,19 | 481.09 | 512.80 | 0.00% | 537.27 | 512.80 | 4.79 | 8.00 |
| | 19,21 | 496.49 | 520.42 | 0.00% | 527.58 | 520.42 | 4.71 | 5.50 |
| E-n33-k4 | 16,22 | 624.89 | 657.93 | 2.78% | 760.81 | 676.19 | 4.66 | - |
| | 16,24 | 641.75 | 666.02 | 0.00% | 666.02 | 666.02 | 8.77 | 747.38 |
| | 19,26 | 608.49 | 680.37 | 0.00% | 743.22 | 680.37 | 8.58 | 26.44 |
| | 22,26 | 619.05 | 680.37 | 0.00% | 690.63 | 680.37 | 8.72 | 6.29 |
| | 24,28 | 635.48 | 670.43 | 0.00% | 670.43 | 670.43 | 9.16 | 17.56 |
| | 25,28 | 602.32 | 650.58 | 0.00% | 650.58 | 650.58 | 8.74 | 158.19 |
| E-n51-k5 | 12,18 | 535.36 | 560.73 | 0.00% | 560.73 | 560.73 | 15.68 | 1007.87 |
| | 12,41 | 545.03 | 564.45 | 0.00% | 598.88 | 564.45 | 15.60 | 208.31 |
| | 12,43 | 546.34 | 564.45 | 0.00% | 607.27 | 564.45 | 16.13 | 288.51 |
| | 39,41 | 663.72 | 690.12 | 9.17% | 753.40 | 753.40 | 15.74 | - |
| | 40,41 | 733.96 | 767.01 | 1.10% | 775.47 | 775.47 | 15.76 | - |
| | 40,43 | 719.81 | 746.89 | 7.50% | 802.91 | 802.91 | 15.87 | - |

Table 3.4: Detailed lower and upper bound results for data set 3. A "-" in time indicates the instance was not solved within 10000 CPU seconds, and a "-" in gap and solution means no solution was found.

varies with the fastest instance solved in less than 10 seconds and 11 out of the 12 instances solved within an hour. On the final instance the gap is 2.78%. For the 50 customer instances the solution time varies a lot, ranging from 15 seconds to no solution within the time limit. The time spent in the heuristic is typically below 50 second and never more than 150 seconds. The heuristic is able to find 19 solutions that are proved optimal by the exact method and on 60 instances the heuristic solution is not improved by the branch-and-cut algorithm.

### 3.4.3 Results for cut separation

In Tables 3.6, 3.7, and 3.8 the detailed results of cut separation is shown. The tables list the instance name (column "Instance"), the number of satellites (column "Sats"), and the number of branch tree nodes (column "Nodes"). For each type of constraints (3.27), (3.33), (3.35), and the capacity inequalities (3.32) and other cuts based on the CVRPSEP package (column "CVRP cuts") we list the number of cuts separated (sub-column "Num") and the separation time (column "Time"). The last column (column "Total") presents the total number of cuts separated and the total running time of the branch-and-cut algorithm.

In general we separate more of the CVRP cuts than the three remaining cut types. The separation time for the CVRP cuts varies from about 5% to about 25% of the total running time, with a single high spike of about 50% on the unsolved 32 customer instance. Of the remaining three types of constraints it is clearly the separation of (3.33) that is most time consuming

| Instance | Sats | DSat | DCust | Root LB | Best LB | Gap | HUB | UB | HTime | BACTime |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | Random | Random | 1432.03 | 1542.88 | 14.83% | 1771.74 | 1771.74 | 7.10 | - |
| 2 | 2 | Random | Random | 1315.86 | 1438.33 | 0.00% | 1438.33 | 1438.33 | 8.53 | 1146.74 |
| 3 | 2 | Sliced | Random | 1393.27 | 1545.54 | 14.49% | 1769.47 | 1769.47 | 7.05 | - |
| 4 | 2 | Sliced | Random | 1283.63 | 1411.55 | 0.89% | 1424.04 | 1424.04 | 8.42 | - |
| 5 | 2 | Forbidden | Random | 1878.35 | 2185.26 | 0.72% | 2201.03 | 2201.03 | 7.11 | - |
| 6 | 2 | Forbidden | Random | 1244.60 | 1279.87 | 0.00% | 1279.87 | 1279.87 | 8.19 | 4463.43 |
| 7 | 2 | Random | Centroids | 1311.82 | 1436.59 | 15.93% | 1665.49 | 1665.49 | 6.94 | - |
| 8 | 2 | Random | Centroids | 1251.32 | 1363.74 | 0.00% | 1364.20 | 1363.74 | 8.46 | 1164.53 |
| 9 | 2 | Sliced | Centroids | 1302.61 | 1431.49 | 15.43% | 1652.36 | 1652.36 | 6.96 | - |
| 10 | 2 | Sliced | Centroids | 1385.47 | 1407.64 | 0.00% | 1422.42 | 1407.64 | 8.36 | 3933.09 |
| 11 | 2 | Forbidden | Centroids | 1836.80 | 2035.77 | 1.46% | 2065.45 | 2065.45 | 6.93 | - |
| 12 | 2 | Forbidden | Centroids | 1157.00 | 1209.42 | 0.00% | 1209.42 | 1209.42 | 8.33 | 22.25 |
| 13 | 2 | Random | Quadrants | 1358.63 | 1464.14 | 13.20% | 1657.47 | 1657.47 | 6.94 | - |
| 14 | 2 | Random | Quadrants | 1270.52 | 1392.62 | 1.70% | 1416.28 | 1416.28 | 8.46 | - |
| 15 | 2 | Sliced | Quadrants | 1366.91 | 1473.70 | 13.06% | 1666.10 | 1666.10 | 7.18 | - |
| 16 | 2 | Sliced | Quadrants | 1263.78 | 1389.17 | 0.00% | 1389.17 | 1389.17 | 8.47 | 1045.12 |
| 17 | 2 | Forbidden | Quadrants | 1789.96 | 2086.66 | 0.46% | 2096.15 | 2096.15 | 7.00 | - |
| 18 | 2 | Forbidden | Quadrants | 1184.69 | 1227.61 | 0.00% | 1227.61 | 1227.61 | 8.35 | 8130.09 |
| 19 | 3 | Random | Random | 1285.92 | 1557.13 | 10.29% | 1717.35 | 1717.35 | 7.53 | - |
| 20 | 3 | Random | Random | 1186.54 | 1265.70 | 17.28% | 1484.45 | 1484.45 | 8.38 | - |
| 21 | 3 | Sliced | Random | 1396.89 | 1570.84 | 7.24% | 1684.51 | 1684.51 | 14.16 | - |
| 22 | 3 | Sliced | Random | 1104.34 | 1281.83 | 0.00% | 1346.99 | 1281.83 | 8.80 | 8636.73 |
| 23 | 3 | Forbidden | Random | 1502.90 | 1626.18 | 11.14% | 1807.35 | 1807.35 | 7.44 | - |
| 24 | 3 | Forbidden | Random | 1204.16 | 1282.68 | 0.00% | 1305.64 | 1282.68 | 8.56 | 6559.87 |
| 25 | 3 | Random | Centroids | 1274.89 | 1503.88 | 2.02% | 1534.21 | 1534.21 | 7.24 | - |
| 26 | 3 | Random | Centroids | 1109.76 | 1167.46 | 0.00% | 1324.17 | 1167.46 | 8.58 | 66.39 |
| 27 | 3 | Sliced | Centroids | 1280.74 | 1467.45 | 8.00% | 1584.84 | 1584.84 | 14.03 | - |
| 28 | 3 | Sliced | Centroids | 1049.89 | 1210.44 | 0.00% | 1259.63 | 1210.44 | 8.81 | 2045.96 |
| 29 | 3 | Forbidden | Centroids | 1386.70 | 1708.58 | 1.68% | 1737.27 | 1737.27 | 7.32 | - |
| 30 | 3 | Forbidden | Centroids | 1116.33 | 1211.59 | 0.00% | 1232.90 | 1211.59 | 8.75 | 17.35 |
| 31 | 3 | Random | Quadrants | 1345.72 | 1468.40 | 11.37% | 1635.36 | 1635.36 | 7.22 | - |
| 32 | 3 | Random | Quadrants | 1042.18 | 1192.81 | 3.59% | 1256.05 | 1235.58 | 8.56 | - |
| 33 | 3 | Sliced | Quadrants | 1247.14 | 1488.99 | 6.62% | 1587.57 | 1587.57 | 7.27 | - |
| 34 | 3 | Sliced | Quadrants | 1056.51 | 1233.23 | 0.06% | 1264.27 | 1233.92 | 8.67 | - |
| 35 | 3 | Forbidden | Quadrants | 1362.81 | 1696.98 | 1.56% | 1723.54 | 1723.54 | 7.39 | - |
| 36 | 3 | Forbidden | Quadrants | 1054.39 | 1228.89 | 0.00% | 1229.61 | 1228.89 | 8.42 | 2038.24 |
| 37 | 5 | Random | Random | 1262.60 | 1484.41 | 11.91% | 1661.16 | 1661.16 | 35.05 | - |
| 38 | 5 | Random | Random | 965.66 | 1167.33 | 8.41% | 1265.49 | 1265.49 | 14.98 | - |
| 39 | 5 | Sliced | Random | 1286.12 | 1515.12 | 0.38% | 1618.25 | 1520.92 | 37.67 | - |
| 40 | 5 | Sliced | Random | 977.40 | 1168.67 | 3.65% | 1211.36 | 1211.36 | 12.29 | - |
| 41 | 5 | Forbidden | Random | 1445.21 | 1644.80 | 9.58% | 1802.39 | 1802.39 | 142.96 | - |
| 42 | 5 | Forbidden | Random | 995.26 | 1179.89 | 14.24% | 1347.93 | 1347.93 | 14.88 | - |
| 43 | 5 | Random | Centroids | 1258.73 | 1414.83 | 15.32% | 1631.52 | 1631.52 | 46.86 | - |
| 44 | 5 | Random | Centroids | 840.00 | 1045.13 | 0.00% | 1144.15 | 1045.13 | 14.07 | 144.01 |
| 45 | 5 | Sliced | Centroids | 1190.54 | 1435.96 | 9.63% | 1574.22 | 1574.22 | 24.45 | - |
| 46 | 5 | Sliced | Centroids | 843.07 | 1077.78 | 2.72% | 1107.04 | 1107.04 | 11.11 | - |
| 47 | 5 | Forbidden | Centroids | 1324.93 | 1571.51 | 10.76% | 1740.64 | 1740.64 | 122.29 | - |
| 48 | 5 | Forbidden | Centroids | 931.71 | 1082.20 | 0.00% | 1252.82 | 1082.20 | 19.27 | 133.42 |
| 49 | 5 | Random | Quadrants | 1209.41 | 1404.52 | 10.76% | 1555.72 | 1555.72 | 33.87 | - |
| 50 | 5 | Random | Quadrants | 843.81 | 1064.26 | 7.29% | 1141.79 | 1141.79 | 21.18 | - |
| 51 | 5 | Sliced | Quadrants | 1150.05 | 1333.37 | 11.82% | 1490.99 | 1490.99 | 37.05 | - |
| 52 | 5 | Sliced | Quadrants | 911.73 | 1113.49 | 1.33% | 1128.33 | 1128.33 | 10.76 | - |
| 53 | 5 | Forbidden | Quadrants | 1313.88 | 1547.35 | 10.87% | 1715.60 | 1715.61 | 125.38 | - |
| 54 | 5 | Forbidden | Quadrants | 994.67 | 1117.24 | 11.60% | 1246.86 | 1246.86 | 16.47 | - |

Table 3.5: Detailed lower and upper bound results for data set 4 (50 customers in all instances). A "-" in time indicates the instance was not solved within 10000 CPU seconds.

with a maximum time of almost 1250 seconds (11% of the total running time) spent on separating the constraint in instance 40 in data set 4. The number of branch tree nodes for the three data sets range from 4 to about 60000 and there is a correlation between the computation time, the number of cuts separated and the number of nodes in the search tree.

## 3.5 Conclusion

We have presented a new exact algorithm for the symmetric 2E-CVRP. The proposed method is a branch-and-cut algorithm based on a non-trivial mathematical model that provides a tight lower bound for the symmetric 2E-CVRP, a feasibility problem to test if integer solutions of the proposed model are valid for the symmetric 2E-CVRP, and a specialized branching scheme to branch on infeasible integer solutions. We have shown that three of the four sets of constraints of exponential size in the proposed model can be separated optimally in polynomial time, and that the last set of constraints are $\mathcal{NP}$-hard to separate. Computational results indicate that our algorithm is superior to the algorithms previously proposed in the literature and we are able to solve 47 out of 93 test instances to optimality where 34 of these are solved to optimality for the first time. The integrality gaps for the branch-and-cut algorithm presented are in some cases still very large so it may well be, that future research in exact algorithms for the 2E-CVRP would be a column generation approach. Column generation based algorithms have in recent years performed very good on routing problems, see e.g., Jepsen et al. [19], Desaulniers [15], Baldacci et al. [5, 7], however the coupling between the echelons in the 2E-CVRP does pose a challenge to incorporate.

## Acknowledgments

We would like to thank David Pisinger, Roberto Roberti and the two anonymous referees for their valuable comments and suggestions which has greatly improved the quality of the paper.

| Customer | Demand |
|---|---|
| 1 | 25 |
| 2 | 25 |
| 3 | 35 |
| 4 | 40 |
| 5 | 40 |
| 6 | 35 |

(a)

| Satellite | Handling costs |
|---|---|
| 1 | 1 |
| 2 | 100 |
| 3 | 1 |

(b)

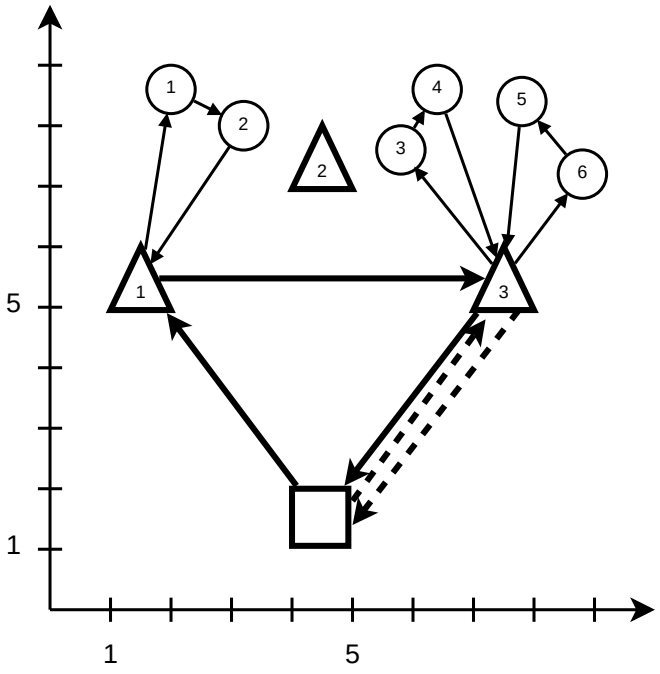



Figure 3.1: An example of a solution to a 2E-CVRP instance. Table (a) and (b) are the customer demands and satellite handling costs.

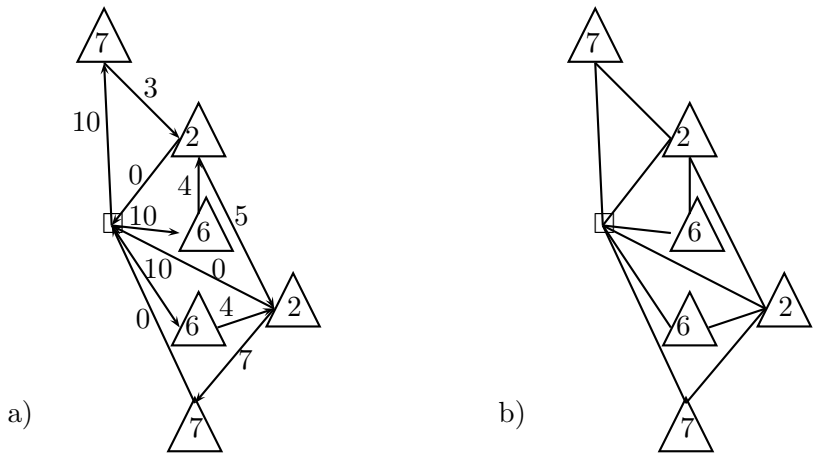



Figure 3.2: Example of a feasible solution to the first echelon of a) [35] and b) model R that cannot be mapped to a feasible 2E-CVRP solution. The capacity of the vehicles are 10 and number of vehicles to use is 3, freight delivered to each satellite is given in the triangles and the variables corresponding to the shown edges all have value 1.

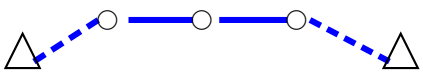


Figure 3.3: Illustration of customer connection to satellites. Triangles are satellites and circles are customers, dashed lines are edges included in constraints (3.33) together with the white customers. Blue edges have weight 1 and red edges have weight 0.

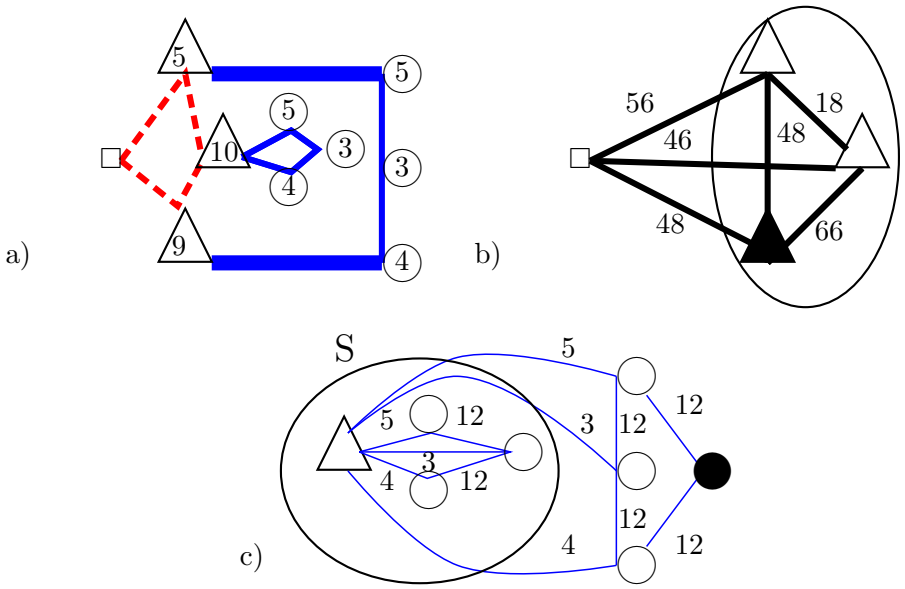



Figure 3.4: In a) the box is the depot, the triangles are the satellites and the circles are the customers. The freight delivered in the satellites and the demand of the customers are given within the triangles and circles. The dashed edges are first echelon and the solid edges are second echelon edges. The capacity for respectively the first and second echelon vehicles are 18 and 12 and the fleet size is 2 and 3, respectively. Unless otherwise shown edge weights are 1. In b) the separation graph for inequalities (3.27) with the bold triangle (satellite) as source is shown ($M = 24$). The set included with the source node $s$ is shown in the ellipse and the value of the cut is 150. c) shows the separation graph for the inequalities (3.35). The triangle represents the selected satellite, the bold circle is the target and the circles are the customers. The violated cut of value 12 is shown as the set $S$.

| Instance | Sats | Nodes | (3.27) | | (3.33) | | (3.35) | | CVRP Cuts | | Total | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Num | Time | Num | Time | Num | Time | Num | Time | Num | Time |
| E-n22-k4 | 6,17 | 4 | 1 | 0.00 | 8 | 0.01 | 0 | 0.00 | 93 | 0.06 | 102 | 0.21 |
| | 8,14 | 14 | 1 | 0.00 | 9 | 0.04 | 1 | 0.01 | 171 | 0.06 | 182 | 1.01 |
| | 9,19 | 387 | 1 | 0.00 | 234 | 0.99 | 25 | 0.05 | 2576 | 2.09 | 2836 | 12.39 |
| | 10,14 | 23 | 1 | 0.00 | 27 | 0.09 | 1 | 0.02 | 171 | 0.12 | 200 | 1.16 |
| | 11,12 | 59 | 1 | 0.00 | 38 | 0.33 | 5 | 0.01 | 279 | 0.22 | 323 | 3.19 |
| | 12,16 | 48 | 1 | 0.01 | 35 | 0.20 | 2 | 0.03 | 446 | 0.16 | 484 | 1.98 |
| E-n33-k4 | 1,9 | 1321 | 1 | 0.04 | 795 | 3.54 | 147 | 0.26 | 6051 | 8.98 | 6994 | 49.42 |
| | 2,13 | 868 | 1 | 0.01 | 671 | 2.55 | 88 | 0.17 | 4689 | 3.28 | 5449 | 34.17 |
| | 3,17 | 13360 | 1 | 0.44 | 8054 | 37.45 | 1278 | 3.36 | 45593 | 286.21 | 54926 | 1126.84 |
| | 4,5 | 748 | 1 | 0.03 | 648 | 0.99 | 90 | 0.21 | 4618 | 4.39 | 5357 | 54.89 |
| | 7,25 | 1846 | 1 | 0.04 | 978 | 6.92 | 136 | 0.49 | 10050 | 12.71 | 11165 | 87.46 |
| | 14,22 | 13 | 1 | 0.00 | 10 | 0.03 | 1 | 0.00 | 377 | 0.19 | 389 | 2.40 |
| E-n51-k5 | 2,17 | 24478 | 1 | 1.83 | 8819 | 281.47 | 776 | 14.08 | 279736 | 2277.34 | 289332 | - |
| | 4,46 | 19 | 1 | 0.00 | 0 | 0.03 | 0 | 0.03 | 1019 | 1.33 | 1020 | 13.25 |
| | 6,12 | 27228 | 1 | 2.94 | 7721 | 340.04 | 553 | 17.66 | 267320 | 1983.33 | 275595 | - |
| | 11,19 | 299 | 1 | 0.01 | 0 | 0.01 | 1 | 0.22 | 6282 | 8.65 | 6284 | 213.61 |
| | 27,47 | 30829 | 1 | 2.78 | 6881 | 300.80 | 1281 | 19.96 | 232064 | 2057.26 | 240227 | - |
| | 32,37 | 8688 | 1 | 0.46 | 2100 | 66.08 | 571 | 4.18 | 70128 | 276.81 | 72800 | 2113.98 |
| E-n51-k5 | 2,4,17,46 | 117 | 4 | 0.01 | 30 | 0.93 | 1 | 0.16 | 3178 | 4.52 | 3213 | 84.03 |
| | 6,12,32,37 | 6956 | 39 | 0.40 | 1850 | 78.33 | 264 | 11.31 | 78166 | 229.74 | 80319 | 3642.79 |
| | 11,19,27,47 | 1715 | 4 | 0.15 | 403 | 19.79 | 87 | 2.91 | 19666 | 34.08 | 20160 | 798.71 |

Table 3.6: Detailed cut separation results for set 2. A "-" in time indicates the instance was not solved within 10000 CPU seconds.

| Instance | Sats | Nodes | (3.27) | | (3.33) | | (3.35) | | CVRP Cuts | | Total | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Num | Time | Num | Time | Num | Time | Num | Time | Num | Time |
| E-n22-k4 | 13,14 | 99 | 1 | 0.01 | 73 | 0.30 | 10 | 0.01 | 309 | 0.20 | 393 | 3.20 |
| | 13,16 | 74 | 1 | 0.00 | 69 | 0.16 | 8 | 0.00 | 247 | 0.18 | 325 | 2.32 |
| | 13,17 | 10 | 1 | 0.00 | 3 | 0.02 | 0 | 0.00 | 87 | 0.12 | 91 | 1.07 |
| | 14,19 | 2449 | 1 | 0.07 | 1421 | 4.56 | 206 | 0.34 | 5498 | 9.20 | 7126 | 61.19 |
| | 17,19 | 246 | 1 | 0.01 | 168 | 0.65 | 16 | 0.00 | 525 | 0.69 | 710 | 8.00 |
| | 19,21 | 95 | 1 | 0.01 | 73 | 0.28 | 9 | 0.01 | 338 | 0.31 | 421 | 5.50 |
| E-n33-k4 | 16,22 | 47021 | 1 | 2.71 | 23057 | 144.62 | 3489 | 14.67 | 106427 | 5780.84 | 132974 | - |
| | 16,24 | 6242 | 1 | 0.20 | 2067 | 14.42 | 235 | 1.59 | 22682 | 268.47 | 24985 | 747.38 |
| | 19,26 | 291 | 1 | 0.01 | 150 | 1.54 | 26 | 0.11 | 1158 | 2.01 | 1335 | 26.44 |
| | 22,26 | 59 | 1 | 0.00 | 15 | 0.20 | 4 | 0.03 | 319 | 0.57 | 339 | 6.29 |
| | 24,28 | 218 | 1 | 0.02 | 93 | 0.66 | 2 | 0.03 | 744 | 1.80 | 840 | 17.56 |
| | 25,28 | 2915 | 1 | 0.11 | 1412 | 8.78 | 107 | 0.52 | 9762 | 34.50 | 11282 | 158.19 |
| E-n51-k5 | 12,18 | 2860 | 1 | 0.05 | 153 | 17.47 | 23 | 1.43 | 10285 | 91.18 | 10462 | 1007.87 |
| | 12,41 | 300 | 1 | 0.01 | 0 | 0.00 | 0 | 0.25 | 1411 | 8.18 | 1412 | 208.31 |
| | 12,43 | 411 | 1 | 0.00 | 0 | 0.01 | 0 | 0.20 | 2454 | 14.69 | 2455 | 288.51 |
| | 39,41 | 16937 | 1 | 0.94 | 940 | 169.47 | 4 | 11.34 | 55803 | 1788.57 | 56748 | - |
| | 40,41 | 17679 | 1 | 0.93 | 9 | 0.82 | 9 | 12.73 | 70609 | 1827.88 | 70628 | - |
| | 40,43 | 18676 | 1 | 0.96 | 2745 | 181.42 | 25 | 11.31 | 59865 | 1567.44 | 62636 | - |

Table 3.7: Detailed cut separation results for set 3. A "-" in time indicates the instance was not solved within 10000 CPU seconds.

| Instance | Sats | Nodes | (3.27) | | (3.33) | | (3.35) | | CVRP Cuts | | Total | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Num | Time | Num | Time | Num | Time | Num | Time | Num | Time |
| 1 | 2 | 25511 | 1 | 1.97 | 5627 | 477.74 | 2358 | 20.72 | 38145 | 419.68 | 46131 | - |
| 2 | 2 | 1778 | 1 | 0.15 | 106 | 23.09 | 217 | 2.26 | 8001 | 64.22 | 8325 | 1146.74 |
| 3 | 2 | 26326 | 1 | 2.20 | 5142 | 426.01 | 2569 | 19.62 | 42419 | 481.41 | 50131 | - |
| 4 | 2 | 35751 | 1 | 2.41 | 10474 | 331.07 | 1493 | 22.46 | 38821 | 301.17 | 50789 | - |
| 5 | 2 | 20501 | 1 | 1.42 | 1134 | 376.96 | 358 | 20.02 | 43831 | 601.29 | 45324 | - |
| 6 | 2 | 9830 | 1 | 0.96 | 2971 | 125.84 | 567 | 8.10 | 24370 | 348.58 | 27909 | 4463.43 |
| 7 | 2 | 39821 | 1 | 2.72 | 7543 | 547.42 | 2692 | 25.72 | 48519 | 495.34 | 58755 | - |
| 8 | 2 | 4613 | 1 | 0.18 | 659 | 25.25 | 144 | 2.48 | 20764 | 166.42 | 21568 | 1164.53 |
| 9 | 2 | 29112 | 1 | 1.67 | 9605 | 593.16 | 1151 | 18.21 | 35022 | 290.14 | 45779 | - |
| 10 | 2 | 11325 | 1 | 0.73 | 1959 | 109.76 | 906 | 9.31 | 21748 | 318.88 | 24614 | 3933.09 |
| 11 | 2 | 20510 | 1 | 1.55 | 2946 | 349.03 | 411 | 16.78 | 54166 | 448.76 | 57524 | - |
| 12 | 2 | 190 | 1 | 0.01 | 92 | 1.91 | 23 | 0.13 | 737 | 1.82 | 853 | 22.25 |
| 13 | 2 | 18579 | 1 | 1.80 | 3082 | 393.41 | 1931 | 19.39 | 35042 | 402.42 | 40056 | - |
| 14 | 2 | 21380 | 1 | 2.04 | 1265 | 246.77 | 6373 | 28.10 | 49104 | 566.79 | 56743 | - |
| 15 | 2 | 29399 | 1 | 2.14 | 11768 | 502.16 | 5133 | 21.16 | 61331 | 592.96 | 78233 | - |
| 16 | 2 | 1852 | 1 | 0.10 | 93 | 18.53 | 275 | 2.06 | 8155 | 76.06 | 8524 | 1045.12 |
| 17 | 2 | 14589 | 1 | 1.25 | 1845 | 355.11 | 473 | 16.42 | 29512 | 340.78 | 31831 | - |
| 18 | 2 | 22552 | 1 | 2.06 | 5026 | 234.49 | 3981 | 21.05 | 42780 | 473.20 | 51788 | 8130.09 |
| 19 | 3 | 31886 | 3 | 3.21 | 9094 | 693.75 | 4806 | 42.34 | 37687 | 502.19 | 51590 | - |
| 20 | 3 | 30061 | 2 | 3.47 | 10802 | 433.03 | 5215 | 38.51 | 41218 | 549.75 | 57237 | - |
| 21 | 3 | 36771 | 2 | 3.95 | 17652 | 842.59 | 4585 | 43.95 | 48191 | 774.08 | 70430 | - |
| 22 | 3 | 25752 | 6 | 2.76 | 9822 | 271.65 | 2096 | 29.63 | 41565 | 405.20 | 53489 | 8636.73 |
| 23 | 3 | 31423 | 3039 | 2.99 | 8566 | 704.65 | 3094 | 41.18 | 46440 | 511.91 | 61139 | - |
| 24 | 3 | 18540 | 1 | 1.82 | 7427 | 202.32 | 1949 | 22.38 | 31200 | 306.06 | 40577 | 6559.87 |
| 25 | 3 | 37998 | 2 | 2.85 | 9156 | 771.85 | 2427 | 42.05 | 45137 | 523.79 | 56722 | - |
| 26 | 3 | 348 | 1 | 0.03 | 127 | 2.70 | 25 | 0.30 | 1865 | 4.65 | 2018 | 66.39 |
| 27 | 3 | 27602 | 1 | 2.13 | 7174 | 574.52 | 1021 | 29.44 | 40672 | 375.56 | 48868 | - |
| 28 | 3 | 7382 | 4 | 0.51 | 3599 | 59.14 | 397 | 8.13 | 22091 | 115.25 | 26091 | 2045.96 |
| 29 | 3 | 24134 | 2 | 1.88 | 8809 | 724.79 | 892 | 29.26 | 32435 | 273.26 | 42138 | - |
| 30 | 3 | 88 | 1 | 0.00 | 45 | 0.76 | 18 | 0.09 | 550 | 1.08 | 614 | 17.35 |
| 31 | 3 | 31371 | 2 | 3.36 | 13697 | 524.29 | 7540 | 38.69 | 57876 | 837.99 | 79115 | - |
| 32 | 3 | 31759 | 2 | 3.24 | 12162 | 326.44 | 5796 | 37.61 | 50220 | 723.20 | 68180 | - |
| 33 | 3 | 25601 | 1 | 2.91 | 8456 | 575.06 | 5053 | 34.20 | 43479 | 673.90 | 56989 | - |
| 34 | 3 | 33409 | 6 | 2.66 | 9787 | 297.43 | 5421 | 41.28 | 55604 | 503.67 | 70818 | - |
| 35 | 3 | 32828 | 6 | 3.69 | 13442 | 761.97 | 6210 | 46.28 | 56377 | 612.85 | 76035 | - |
| 36 | 3 | 8459 | 10 | 0.50 | 1886 | 69.15 | 1614 | 10.40 | 22892 | 167.87 | 26402 | 2038.24 |
| 37 | 5 | 19925 | 19 | 2.32 | 6063 | 572.16 | 1590 | 51.74 | 28865 | 953.76 | 36537 | - |
| 38 | 5 | 30449 | 13 | 4.27 | 7933 | 524.13 | 10561 | 87.30 | 49778 | 519.68 | 68285 | - |
| 39 | 5 | 30438 | 28 | 3.87 | 11135 | 743.32 | 5504 | 73.18 | 46662 | 582.89 | 63329 | - |
| 40 | 5 | 42902 | 41 | 9.66 | 21713 | 1222.42 | 26209 | 145.39 | 51632 | 1146.44 | 99595 | - |
| 41 | 5 | 24673 | 13 | 3.63 | 4635 | 654.63 | 2406 | 61.76 | 39095 | 411.45 | 46149 | - |
| 42 | 5 | 38924 | 70 | 5.35 | 17813 | 802.11 | 13092 | 106.59 | 56173 | 715.74 | 87148 | - |
| 43 | 5 | 25052 | 21 | 2.73 | 8215 | 562.04 | 716 | 51.67 | 41492 | 387.39 | 50444 | - |
| 44 | 5 | 469 | 12 | 0.01 | 112 | 11.48 | 221 | 1.56 | 1594 | 9.71 | 1939 | 144.01 |
| 45 | 5 | 35598 | 44 | 3.16 | 7739 | 576.05 | 1980 | 73.34 | 59879 | 723.66 | 69642 | - |
| 46 | 5 | 59943 | 9 | 6.83 | 23421 | 941.94 | 7081 | 119.33 | 26907 | 622.15 | 57418 | - |
| 47 | 5 | 24414 | 66 | 3.07 | 9837 | 663.53 | 2207 | 58.27 | 31847 | 332.18 | 43957 | - |
| 48 | 5 | 598 | 20 | 0.05 | 211 | 18.36 | 356 | 2.20 | 1439 | 10.70 | 2026 | 133.42 |
| 49 | 5 | 16810 | 12 | 2.79 | 8895 | 818.65 | 3177 | 48.27 | 17468 | 218.55 | 29552 | - |
| 50 | 5 | 24534 | 108 | 4.76 | 5947 | 526.31 | 22330 | 110.33 | 41079 | 673.61 | 69464 | - |
| 51 | 5 | 27501 | 38 | 3.33 | 12831 | 680.13 | 4388 | 66.82 | 40037 | 564.73 | 57294 | - |
| 52 | 5 | 52982 | 40 | 7.20 | 16588 | 1064.43 | 22863 | 150.38 | 55437 | 1201.23 | 94928 | - |
| 53 | 5 | 19113 | 41 | 3.03 | 7849 | 545.34 | 2165 | 49.50 | 27148 | 305.86 | 37203 | - |
| 54 | 5 | 20673 | 7 | 2.02 | 8019 | 349.76 | 4022 | 55.25 | 42636 | 406.95 | 54684 | - |

Table 3.8: Detailed cut separation results for set 4 (50 customers in all instances). A "-" in time indicates the instance was not solved within 10000 CPU seconds.

# Bibliography


[1] C. Archetti and M.G. Speranza. The split delivery vehicle routing problem: A survey. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 103–122. Springer, 2008.

[2] C. Archetti, N. Bianchessi, and M.G. Speranza. A column generation approach for the split delivery vehicle routing problem. *Networks*, 2010. Conditionally accepted.

[3] R. Baldacci, M. Dell'Amico, and J. Salazar González. The capacitated $m$-ring-star problem. *Operations Research*, 55(6):1147–1162, 2007.

[4] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 10 2008.

[5] R. Baldacci, E. Bartolini, A. Mingozzi, and R. Roberti. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science*, 7(3):229–268, 2010.

[6] R. Baldacci, A. Mingozzi, and R. Wolfler Calvo. An exact method for the capacitated location-routing problem. *Operations Research*, 2011. Forthcoming.

[7] R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, 2011. Forthcoming.

[8] J.-M. Belenguer, E. Benavent, C. Prins, C. Prodhon, and R. Wolfler Calvo. A branch-and-cut method for the capacitated location-routing problem. *Computers & Operations Research*, 38:931–941, 2011.

[9] J.M. Belenguer, M.C. Martinez, and E. Mota. A lower bound for the split delivery vehicle routing problem. *Operations research*, 48(5):801–810, 2000.

[10] I.M. Chao. A tabu search method for the truck and trailer routing problem. *Computers and Operations Research*, 29(1):33–51, 2002.

[11] C. Contardo, J.-F. Cordeau, and B. Gendron. A branch-and-cut algorithm for the capacitated location-routing problem. Submitted, 2010.

[12] T.G. Crainic, S. Mancini, G. Perboli, and R. Tadei. Clustering-based heuristics for the two-echelon vehicle routing problem. Technical report CIRRELT-2008-46, CIRRELT, Montreal, Canada, 2008.

[13] T.G. Crainic, S. Mancini, G. Perboli, and R. Tadei. Two-echelon vehicle routing problem: A satellite location analysis. *Procedia: Social - Behaivioral Studies*, 2(3):5944–5955, 2010.

[14] T.G. Crainic, S. Mancini, G. Perboli, and R. Tadei. Multi-start heuristics for the two-echelon vehicle routing problem. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6622 LNCS:179–190, 2011.

[15] G. Desaulniers. Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations research*, 58(1):179–192, 2010.

[16] J. Gonzales Feliu, G. Perboli, R. Tadei, and D. Vigo. The two-echelon capacitated vehicle routing problem. Technical report DEIS OR.INGCE 2007/2(R), DEIS, Bologna, Italy, 2007.

[17] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R.F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming, Ser. A*, 106(3):491–511, 2006.

[18] B. Gavish and S.C. Graves. The traveling salesman problem and related problems. Technical Report OR 078-78, Massachusetts Institute of Technology, July 1978.

[19] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511, 2008.

[20] M. Jin, K. Liu, and R.O. Bowden. A two-stage algorithm with valid inequalities for the split delivery vehicle routing problem. *International Journal of Production Economics*, 105:228–242, 2007.

[21] M. Jin, K. Liu, and B. Eksioglu. A column generation approach for the split delivery vehicle routing problem. *Operations Research Letters*, 36: 265–270, 2008.

[22] G. Laporte. What you should know about the vehicle routing problem. *Naval Research Logistics*, 54:811–819, 2007.

[23] G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43 (4):408–416, 2009.

[24] G. Laporte and Y. Nobert. A branch and bound algorithm for the capacitated vehicle routing problem. *OR Spektrum*, 5:77–85, 1983.

[25] G. Laporte, Y. Nobert, and D. Arpin. An exact algorithm for solving a capacitated location-routing problem. *Annals of Operations Research*, 6(9):291–310, 1986.

[26] A.N. Letchford and J.-J. Salazar-González. Projection results for vehicle routing. *Mathematical Programming*, 105(2-3):251–274, 2006.

[27] J. Lysgaard. The cvrpsep package. http://www.hha.dk/ lys/CVRPSEP.htm, 2003.

[28] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423–445, 2004.

[29] F. Margot. Symmetry in integer linear programming. In M. Jünger, T.M. Liebling, D. Naddef, G.L. Nemhauser, W.R. Pulleyblank, G. Reinelt, G. Rinaldi, and L.A. Wolsey, editors, *50 Years of Integer Programming 1958-2008*, pages 647–686. Springer Berlin Heidelberg, 2010. ISBN 978-3-540-68279-0.

[30] L. Moreno, M. Poggi de Aragão, and E. Uchoa. Improved lower bounds for the split delivery vehicle routing problem. *Operations Research Letters*, 38:302–306, 2010.

[31] G. Nagy and S. Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177:649–672, 2007.

[32] V. Nguyen, C. Prins, and C. Prodhon. A multi-start evolutionary local search for the two-echelon location routing problem. In M. Blesa, C. Blum, A. Roli, and M. Sampels, editors, *Lecture Notes in Computer Science*, volume 6373, pages 88–102. Springer, 2010.

[33] G. Perboli, R. Tadei, and F. Masoero. Models and cuts for the two-echelon vehicle routing problem. In *Proceedings of the International Network Optimization Conference 2009*, Pisa, Italy, April 2009.

[34] G. Perboli, R. Tadei, and F. Masoero. New family of valid inequalities for the two-echelon vehicle routing problem. *Electronic Notes in Discrete Mathematics*, 36:639–646, 2010.

[35] G. Perboli, R. Tadei, and D. Vigo. The two-echelon capacitated vehicle routing problem: models and math-based heuristics. *Transportation Science*, 2011. doi: 10.1287/trsc.1110.0368.

[36] D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):2403–2435, 2007.

[37] K.C. Tan, Y.H. Chew, and L.H. Lee. A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, 172:855–885, 2006.

# Chapter 4 A Branch-and-Cut Algorithm for the Capacitated Profitable Tour Problem

Mads Kehlet Jepsen Bjørn Petersen Simon Spoorendonk[1]
David Pisinger
*DTU Management Engineering, Technical University of Denmark*

## Abstract

This paper considers the capacitated profitable tour problem (CPTP). The CPTP belongs to the group of problems known as travelling salesman problems with profits. In CPTP each customer is associated with a profit and a demand and the objective is to find a capacitated tour (rooted in a depot node) that minimizes the total travel distance minus the profit of the visited customers. The CPTP can be recognized as the sub-problem in many column generation applications. We present a branch-and-cut algorithm based on a formulation for the undirected CPTP. Valid inequalities are presented among which we introduce a new family denoted *rounded multistar inequalities* and we prove their validity for the CPTP. Computational experiments are performed on a set instances known from the literature and some newly generated instances. The results indicate that the presented algorithm is highly competitive with the dynamic programming algorithms. In particular, we are able to solve instances with 800 nodes to optimality where the dynamic programming algorithms cannot solve instances with more than 200 nodes. Moreover the two algorithms compliment each other well.

**Keywords:** Branch-and-cut algorithm, valid inequalities, profitable tour problem, capacitated shortest path problem, travelling salesman problem

## 4.1 Introduction

The capacitated profitable tour problem (CPTP) can be defined on a complete undirected graph $G(V, E)$ with nodes $V = N \cup \{0\}$ where $N$ is a set of customers and 0 is the depot node, and $E$ is the set of edges connecting the

[*]Supported by the Danish Council for Independent Research | Technology and Production Sciences (project 274-08-0353)

nodes in $V$. A cost $c_e$ is associated with each edge $e \in E$. Also, a demand $d_i$ and a profit $p_i$ is associated with each customer $i \in N$ and a capacity $Q$ is given for the maximum load of tour. The objective is to find a tour rooted in the depot where the demand accumulated at the customers does not exceed the capacity, and the total travel distance subtracting the profits gained by visiting customers is minimized.

The CPTP is a side-constrained version of the profitable tour problem named by Dell'Amico et al. [12], a problem that falls within the category of traveling salesman problems with profits as classified by Feillet et al. [15]. Other problems in this category are the orienteering problem (OP) (also known as the selective travelling salesman problem) and the prize-collecting travelling salesman problem (PCTSP). In the OP the total tour length is bounded from above, and the objective is to maximize the profit gained by visiting customers. In the PCTSP the objective is similar to the CPTP but a minimum amount of profits must be collected on the tour. In the context of the capacitated vehicle routing problem (CVRP) the CPTP appears as the sub-problem in column generation methods, see e.g. Baldacci et al. [3, 2]. In this context, the CPTP is often transformed to a path problem (a path is obtained from the tour by splitting the depot into two nodes) and is denoted the elementary shortest path problem with resource constraints. The resource is given as an accumulation of demand of the visited customers and is constrained by the capacity. However, in recent routing applications the sub-problem is complicated considerably by the introduction of additional cuts in the column generating master problem, such as the strong capacity inequalities [2], the subset-row inequalities [20], the Chvátal-Gomory rank-1 cuts [28], and the clique inequalities [33]. The sub-problem can no longer be considered a CPTP. Moreover, the sub-problems are often solved as feasibility problems instead of optimization problems which may favour other types of combinatorial algorithms than branch-and-cut algorithms.

Laporte and Martello [21] showed that the OP is $\mathcal{NP}$-hard by reduction from the Hamilton circuit problem. Using a similar reduction it can be shown that the CPTP also belongs to the class of $\mathcal{NP}$-hard problems. If there are no cycles with negative cost in the graph $G$, then the CPTP is solvable in pseudo-polynomial time using a dynamic programming algorithm. In this particular case the CPTP relates to the constrained shortest path problem (again by transformation to a path problem). Several algorithms based on dynamic programming exist for this problem, see e.g., Beasley and Christofides [6],Carlyle et al. [9], Dumitrescu and Boland [13], and Muhandiramge and Boland [25].

Bixby [7] considers the CPTP in her PhD thesis on the CVRP and present a mathematical model and a branch-and-cut (BAC) algorithm. Letchford and Salazar-Gonzalez [24] discuss projection results for the CVRP and present two families of multistar inequalities that are valid for the CPTP. Other work on the CPTP in a CVRP context is mainly concerned with

dynamic programming algorithms. Feillet et al. [14] present a dynamic programming algorithm where the elementarity of the path is ensured by use of an additional resource per node. Chabrier [10] improved on the labeling algorithm by applying various bounding and dominance procedures to avoid the extension of unpromising paths. Christofides et al. [11] proposed a bi-directional labeling algorithm where paths are extended from both ends of the path until half of the capacity is reached. The partial paths are then combined to construct a full path. Righini and Salani [29] generalized this approach to other types of resources. Independently, Boland et al. [8] and Righini and Salani [30] proposed to initially relax the node resources and add them iteratively until the path is elementary. In the former paper this is referred to as a *state space augmentation* algorithm and in the latter it is denoted a *decremental state space relaxation* algorithm. Furthermore, Righini and Salani [30] propose to use the result of the relaxed problem in a branch-and-bound algorithm. Fischetti et al. [16] and Gendreau et al. [18] present BAC algorithms for the OP. They present several valid inequalities, many of which are also valid for the CPTP. Indeed, we prove that the polytope of the CPTP can be transformed to an instance of the polytope for the OP. However, Gendreau et al. [18] also present some inequalities related to the objective function of the OP that are not valid for the CPTP. Bauer et al. [5] consider the cardinality constrained circuit problem (CCCP) where a minimum cost circuit of maximal cardinality in a graph is sought. The CCCP is equivalent to the CPTP with unit demands if one node is fixed in the CCCP (the depot node of the CPTP). Two mathematical models are presented and several valid inequalities are investigated. Bauer et al. [5] suggest to solve the CPTP by a BAC algorithm, but to our knowledge this has not been pursued.

The contribution of this paper is the introduction of an IP model for the CPTP and a BAC algorithm for solving it. This includes the adaption of several valid inequalities from e.g. the OP and the CCCP, the introduction of the *rounded multistar inequalities*, and a proof of validity for all inequalities with regard to the CPTP. Also, we have successfully implemented a separation heuristic for finding knapsack large multistar inequalities that prove their usefulness for the CPTP. The computational experiments show that the BAC algorithm is competitive with the state-of-the-art dynamic programming algorithms. In particular, the BAC algorithm is able to solve instances with 800 nodes to optimality where the dynamic programming algorithms cannot solve instances with more than 200 nodes. In general the two algorithms appear to complement each other well. The BAC algorithm performs best on those instances that are difficult to solve by dynamic programming.

The paper is organized as follows: Section 4.2 contains an integer programming model for the CPTP, Section 4.3 describes the cutting planes used in the BAC algorithm, Section 4.4 presents the separation results for

these cutting planes, the computational results are found in Section 4.5, and Section 4.6 concludes the work.

## 4.2 Mathematical Model

Recall the definition of CPTP on a graph $G(V, E)$. The traversal of an edge $e$ is indicated by the binary variables $x_e$ for all $e \in E$ and a visit to node $i$ is indicated by the binary variable $y_i$ for all $i \in V$. Some short-hand notation: for node set $S$ we use $\delta(S)$ to indicate the edge set consisting of the edges between $S$ and its compliment $\overline{S}$, $E(S)$ to indicate the edge set of the complete sub-graph spanned by $S$, and $E(S : T)$ for $T \cap S = \emptyset$ to indicate the edges connecting $S$ and $T$. For singleton sets we simply write $i$ instead of $\{i\}$, e.g., $\delta(i)$ is the set of edges connected to node $i$.

Without loss of generality it is assumed that there exists no one-customer tours. Such tours can be calculated a priori in $O(N)$ time and provided as valid upper bounds. Let the CPTP be stated as the following integer program

$$\min \sum_{e \in E} c_e x_e - \sum_{i \in N} p_i y_i \tag{4.1}$$

$$\sum_{e \in \delta(i)} x_e = 2y_i \qquad \forall i \in V \tag{4.2}$$

$$y_0 = 1 \tag{4.3}$$

$$\sum_{e \in \delta(S)} x_e \geq 2y_i \qquad \forall i \in S, \forall S \subseteq N, |S| \geq 2 \tag{4.4}$$

$$\sum_{i \in N} d_i y_i \leq Q \tag{4.5}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E \tag{4.6}$$

$$y_i \in \{0, 1\} \qquad \forall i \in V. \tag{4.7}$$

The objective function (4.1) minimizes the overall cost, i.e., the travel cost of traversing the edges on the tour subtracted by profit gained by visiting the customers. Constraints (4.2) are the degree constraints of the nodes and constraint (4.3) ensure that there is exactly one tour going through the depot. Constraints (4.4) known as the generalized subtour elimination constraints (GSEC) eliminates sub-tours by ensuring connectivity of the edges and constraint (4.5) imposes that the capacity on the tour is not exceeded. Note that the sub-tour elimination constraints (4.4) can be equivalently expressed as

$$\sum_{e \in E(S)} x_e \leq \sum_{i \in S \setminus j} y_i \qquad \forall i \in S, \forall S \subseteq N, |S| \geq 2 \tag{4.8}$$

Depending on the size of the set $S$, the constraint (4.4) may contain less variables than (4.8). The sparser formulation is often preferred since it typically leads to a faster solution of the LP relaxation.

The mathematical model contains $O(|E|)$ variables and an exponential amount of constraints due to the sub-tour elimination constraints (4.4).

Let $P_C = \text{conv}\{(x,y) \in \mathbb{R}^{|E|\times|V|} \mid (x,y)\text{satisfies (4.2)} - \text{(4.4) and (4.6)} - \text{(4.7)}\}$ be the circuit polytope. The capacity constraint (4.5) where $d, Q \geq 0$ gives rise to the 0-1 knapsack polytope $P_K = \text{conv}\{x \in \mathbb{R}^{|E|} \mid x\text{satisfies (4.5)}-\text{(4.6)}\}$. We examine the intersection of these polytopes namely the node-capacitated circuit polytope $P_N = \text{conv}\{(x,y) \in \mathbb{R}^{|E|\times|V|} \mid (x,y)\text{satisfies (4.2)}-\text{(4.7)}\}$. As a result all inequalities valid for $P_C$ or $P_K$ are valid for $P_N$.

We will briefly discuss the variant of the CPTP where the demand is associated to the edges instead of the nodes. The capacity constraint (4.5) is then substituted by the constraint

$$\sum_{e\in E} d_e x_e \leq Q \tag{4.9}$$

The polytope of this problem is the edge-capacitated circuit polytope $P_E = \text{conv}\{(x,y) \in \mathbb{R}^{|E|\times|V|} \mid (x,y)\text{satisfies (4.2)} - \text{(4.4)}, \text{(4.6)} - \text{(4.7) and (4.9)}\}$. The polytopes $P_N$ and $P_E$ are very similar and in fact it is possible to transform any $P_N$ instance to an instance of $P_E$ by setting $d_e = 1/2(d_i + d_j)$. The constraint set of the OP is an edge-capacitated circuit polytope and due to the transformation it is natural to adapt valid inequalities for $P_N$ from this problem.

## 4.3 Valid Inequalities

Next we present previously studied valid inequalities for the circuit polytope $P_C$, the knapsack polytope $P_K$, the node-capacitated circuit polytope $P_N$ and the edge-capacitated circuit polytope $P_E$. Furthermore, we present new families of valid inequalities for $P_N$ that approximate the non-linear inequalities that stems from a rounding of various multistar inequalities. The proof of validity with regard to the CPTP is given for all inequalities.

### 4.3.1 The Circuit Polytope

The **logical constraints** are given for each edge $e \in E$ and one of its endpoints $i \in V$ as

$$x_e \leq y_i \qquad \forall e \in \delta(i), \forall i \in N. \tag{4.10}$$

There are $2|E|$ such constraints. The constraints are due to Leifer and Rosenwein [22]. We present a formal proof for their validity for the polytope $P_C$.

**Theorem 1.** The logical constraints (4.10) are valid for $P_C$.

*Proof.* The constraints are a special case of the sub-tour elimination constraints on the form (4.8) with $S = \{i, j\}$ being the endpoints of $e$. This results in the edge set $E(S) = \{e\}$ and by subtracting node $j$ from $S$ we obtain (4.10). □

### Cut Inequalities

Seymour [31] introduced the cut inequalities

$$\sum_{e \in \delta(S) \setminus f} x_e - x_f \geq 0 \qquad \forall S \subseteq N, |S| \geq 2, \forall f \in \delta(S) \tag{4.11}$$

when studying the circuit polytope. Bixby [7] denote the inequalities the co-circuit inequalities in her study of the node-capacitated circuit polytope.

**Theorem 2.** The cut inequalities (4.11) are valid for $P_C$.

*Proof.* Consider a sub-tour elimination constraint (4.4) with node set $S$ and node $i \in S$ on the right-hand side. Subtract the logical constraint (4.10) for edge $f$ with one endpoint being node $i$ to obtain the cut inequalities (4.11). □

For our definition of $P_C$ using constraints (4.2)-(4.7) we obtain the following result:

**Theorem 3.** The cut inequalities (4.11) are dominated by the sub-tour elimination constraints (4.4) for the polytope $P_C$ when the logical constraints (4.10) hold for all edges.

*Proof.* Consider a constraint of type (4.11). Let $f$ have end-points $i \in V \setminus S$ and $j \in S$ then add the corresponding degree constraint (4.2) to inequality (4.11) getting:

$$\sum_{e \in \delta(S)} x_e + \sum_{e \in \delta(j) \setminus f} x_e - x_f \geq 2y_j. \tag{4.12}$$

When $x_f \leq y_j$ (the logical constraint) then $\sum_{e \in \delta(j) \setminus f} x_e - x_f \geq 0$. It follows that the left-hand side of (4.12) is always greater than or equal to the left-hand side of the sub-tour elimination constraints (4.4) expressed for node $j$, hence (4.11) are dominated by (4.4). □

### 2-Matching Inequalities

Fischetti et al. [16], Gendreau et al. [18], and Bauer [4] presented the 2-matching inequalities adapted from the TSP polytope as

$$\sum_{e \in H} x_e + \sum_{e \in T_i} x_e \leq \sum_{i \in H} y_i + \frac{1}{2}\left(|T| - 1\right) \tag{4.13}$$

for all $H \subset V$ and all $T \subset E$ satisfying

(i) $e \in \delta(H)$

(ii) All pair of edges $e, f \in T$ have disjoint endpoints

(iii) $|T| \geq 3$ and odd

A proof of validity for the circuit polytope is

**Theorem 4** (Gendreau et al. [18])**.** The 2-matching inequalities (4.13) are valid for $P_C$.

*Proof.* Sum up the degree constraints (4.2) for all $i \in H$ to obtain

$$2\sum_{e \in H} x_e + \sum_{e \in \delta(H)} x_e = 2\sum_{i \in H} y_i. \tag{4.14}$$

Since $x_e \leq 1$ for all $e \in T$ adding these constraints to (4.14) and divide by 2 yields

$$\sum_{e \in H} x_e + \sum_{e \in T} x_e \leq \sum_{i \in H} y_i + \frac{1}{2}|T|. \tag{4.15}$$

As the variable terms of (4.15) is integer and $|T| \geq 3$ and is odd, rounding down on the right hand side yields (4.13). □

### Comb Inequalities

Bauer [4] adapts the comb inequalities from the TSP polytope to obtain

$$\sum_{e \in E(H)} x_e + \sum_{i=1}^{t} \sum_{e \in E(T_i)} x_e \leq \sum_{j \in H} y_j + \sum_{i=1}^{t} \sum_{j \in T_i} y_j - \sum_{i \in U} y_i - \sum_{i \in R} y_i + \frac{1}{2}\left(t - 1\right) \tag{4.16}$$

for all $H \subset V$ and all $T_i \subset E$, for $i = 1, \ldots, t$ satisfying

(i) $|T_i \cap H| \geq 1, i = 1, \ldots, t$

(ii) $|T_i \setminus H| \geq 1, i = 1, \ldots, t$

(iii) $T_i \cap T_j = \emptyset, 1 \leq i < j \leq t$

(iv) $|E| \geq 3$ and odd

The set $H$ is called the handle and sets $T_i$ are called the teeth of the comb. For each tooth choose a node that is not belonging to any handle and let that set be denoted the set of root nodes $R$. For $1i \leq t$ and $H \cap T_i \neq \emptyset$ choose a node $u_i \in H \cap T_i$. We denote $U = \cap_{i=1}^{t} u_i$ as the links. If (i) is satisfied with equality the comb inequality is called *simple*. Furthermore, if (ii) is also satisfied by equality it implies $|T_i| = 2$ for $i = 1, \ldots, t$ and the resulting inequality is a 2-matching inequality (4.13).

Bauer [4] shows the validity of the comb inequalities by showing the validity of the super-set of clique trees inequalities. We provide an alternative (and simpler) proof of validity only for the comb inequalities.

**Theorem 5.** The comb inequalities (4.16) are valid for $P_C$.

*Proof.* Sum up the degree constraints (4.2) for all $i \in H$ to obtain

$$2 \sum_{e \in E(H)} x_e + \sum_{e \in \delta(H)} x_e = \sum_{j \in H} y_j.$$

Add the sub-tour elimination constraints (4.8) for each tooth $T_i$, $i = 1, \ldots t$ and let the subtracted node on the right-hand side be node $y_u$ for $u \in H \cap T_i$. Denote the set of all these nodes $U$. We obtain

$$2 \sum_{e \in E(H)} x_e + \sum_{e \in \delta(H)} x_e + \sum_{i=1}^{t} \sum_{e \in E(T_i)} x_e \leq 2 \sum_{j \in H} y_j + \sum_{i=1}^{t} \sum_{j \in T_i} y_j - \sum_{j \in U} y_j.$$

Add (4.8) for each intersection between the handle and a tooth $H \cap T_i$ with the subtracted node $y_u$ already in $U$, and add up for each set $T_i \setminus H$ with the subtracted node $y_r$. The set of the latter nodes are denoted $R$. We obtain

$$2 \sum_{e \in E(H)} x_e + \sum_{i=1}^{t} \sum_{e \in \delta(H) \setminus \delta(T_i)} x_e + 2 \sum_{i=1}^{t} \sum_{e \in E(T_i)} x_e \leq 2 \sum_{j \in H} y_j + 2 \sum_{i=1}^{t} \sum_{j \in T_i} y_j - 2 \sum_{u \in U} y_j - \sum_{j \in R} y_j.$$

For each tooth add the constraint $y_r \leq 1$ for the node $r \in R \cap T_i$ which sums up to $\sum_{i \in R} y_i \leq t$.

$$2 \sum_{e \in E(H)} x_e + \sum_{i=1}^{t} \sum_{e \in \delta(H) \setminus \delta(T_i)} x_e + 2 \sum_{i=1}^{t} \sum_{e \in E(T_i)} x_e \leq 2 \sum_{j \in H} y_j + 2 \sum_{i=1}^{t} \sum_{j \in T_i} y_j - 2 \sum_{u \in U} y_j - 2 \sum_{j \in R} y_j + t.$$

Dividing by two and rounding down all coefficients leads to

$$\sum_{e \in E(H)} x_e + \sum_{i=1}^{t} \sum_{e \in E(T_i)} x_e \leq \sum_{j \in H} y_j + \sum_{i=1}^{t} \sum_{j \in T_i} y_j - \sum_{u \in U} y_j - \sum_{j \in R} y_j + \frac{1}{2} t. \tag{4.17}$$

As the variable terms of (4.17) is integer and $t \geq 3$ is odd, rounding down on the right hand side yields (4.16). □

**Clique Tree Inequalities**

Bauer [4] adapts the clique tree inequalities from the TSP polytope. Let a clique tree be a connected sub-graph of $G$ composed of cliques which satisfy the following properties

(i) The cliques are partitioned into two sets, teeth $T_i, \ldots, T_t$ and handles $H_1, \ldots, H_h$.

(ii) No two teeth intersect, i.e., $T_i \cap T_j = \emptyset, 1 \leq i < j \leq t$

(iii) No two handles intersect, i.e., $H_i \cap H_j = \emptyset, 1 \leq i < j \leq h$

(iv) Each tooth contains at least two nodes and at most $|V| - 2$ nodes and at least one node not belonging to any handle, i.e., $2 \leq |T_i| \leq |V| - 2, T_i \setminus \bigcup_{j=1}^{h} H_j \neq \emptyset, 1 \leq i \leq t$.

(v) For each handle, the number of teeth intersecting it is odd and at least three.

(vi) If a tooth and a handle intersect, then their intersection is an articulation set of the clique tree.

For each tooth choose a node that is not belonging to any handle, and let that set be denoted the set of root nodes $R$. For every non-empty intersection $H_i \cap T_j, 1 \leq i \leq h, 1 \leq i \leq t$ choose a node, and denote it the set of links $U$.

$$\sum_{i=1}^{h} \sum_{e \in E(H_i)} x_e + \sum_{i=1}^{t} \sum_{e \in E(T_i)} x_e \leq \sum_{i=1}^{h} \sum_{j \in H_i} y_j + \sum_{i=1}^{t} \sum_{j \in T_i} y_j - \sum_{i \in U} y_i - \sum_{i \in R} y_i + \frac{1}{2}(t-1) \tag{4.18}$$

for all $H_i \subset V, i = 1, \ldots, h$ and all $T_i \subset V, i = 1, \ldots, t$. When $h = 1$ the inequality is a comb inequality.

Bauer [4] prove the validity of (4.18) for $P_C$. Here we provide an alternative proof similar to Grötschel and Pulleyblank [19]. The proof is adapted from the proof of the corresponding clique tree inequalities for the TSP. We start with a few remarks resulting from the effect of *gluing* and *splitting* clique trees into other clique trees. Those are operation that allow merging (gluing) two existing clique tree into a new clique tree and dividing (splitting) an existing clique tree into two smaller clique trees, see Grötschel and Pulleyblank [19] for details on the gluing and splitting operations. Let $s(C)$ denote the *size* of a clique tree $C$ where $s(C)$ is equal to the right-hand side of (4.18). Remark 3.6 (c) in [19] for $P_C$ is equivalent to

**Remark 1.** Let $C$ be a clique tree and $H$ a handle of $C$ intersecting $k$ teeth. Let $C_i, \ldots, C_k$ be the clique trees obtained from $C$ by splitting into at the

handle $H$. Then

$$\sum_{i=1}^{k} s(C_i) = s(C) - \sum_{i \in H} y_i + \sum_{i \in H \cap U} y_i + \frac{1}{2}(k-1)$$

**Theorem 6.** The clique tree inequalities (4.18) are valid for $P_C$.

*Proof.* Consider a clique tree with handles $H_1, \ldots, H_h$ and teeth $T_1, \ldots, T_t$. As in [19] the proof is by induction on the number of handles. If the clique tree has no handle then the clique tree inequality is a sub-tour elimination constraint (4.8) and the proof is done.

Suppose the theorem holds for all clique trees with $h$ handles, and assume that $C$ is a clique tree with $h+1$ handles. Choose a handle of $C$ and denote the other handles $H_1, \ldots, H_h$. Let $T_1, \ldots, T_k$ be the teeth intersecting $H$ and let $C_i, \ldots, C_k$ be the clique trees obtained from $C$ by splitting at a handle $H$. All of these clique trees have at most $h$ handles. Assume that $C_i$ contains $T_i, i = 1, \ldots, k$, and let $a_i^\intercal x \leq s(C_i)$ be shorthand for these clique tree inequalities. For every $C_i, i = 1, \ldots, k$, let $\overline{C}_i$ be the clique tree obtained by replacing $T_i$ with $T_i \setminus H$, and let $\overline{a}_i^\intercal x \leq s(\overline{C}_i)$ be the corresponding clique tree inequality. By Remark 1 we have

$$\sum_{i=1}^{k} s(C_i) = s(C) - \sum_{i \in H} y_i + \sum_{i \in H \cap U} y_i + \frac{1}{2}(k-1)$$

which implies

$$\sum_{i=1}^{k} s(\overline{C}_i) = s(C) - \sum_{i \in H} y_i + \sum_{i \in H \cap U} y_i - \sum_{i=1}^{k} (H \cap T_i) + \frac{1}{2}(k-1).$$

From this we obtain (setting $H_{h+1} = H$)

$$\begin{aligned}
&2\left(\sum_{i=1}^{h} \sum_{e \in E(H_i)} x_e + \sum_{i=1}^{t} \sum_{e \in E(T_i)} x_e\right) && \leq \\
&\sum_{i=1}^{k} \left(a_i^\intercal x + \overline{a}_i^\intercal x + \sum_{e \in E(H \cap T_i)} x_e\right) + \sum_{i \in H} \sum_{e \in E(i:V \setminus \{i\})} x_e && \leq \\
&\sum_{i=1}^{k} \left(s(C_i) + s(\overline{C}_i) + \sum_{i \in H \cap T_i} y_i - \sum_{i \in U \cap H \cap T_i} y_i\right) + 2\sum_{i \in H} y_i && = \\
&\sum_{i=1}^{k} s(C_i) + \sum_{i=1}^{k} s(\overline{C}_i) + \sum_{i=1}^{k} \sum_{i \in H \cap T_i} y_i - \sum_{i \in U \cap H} y_i + 2\sum_{i \in H} y_i && = \\
&2s(C) + \sum_{i \in U \cap H} y_i - k + 1.
\end{aligned}$$

Since $\sum_{i \in U \cap H} y_i \leq k$ we obtain

$$2\left(\sum_{i=1}^{h}\sum_{e \in E(H_i)} x_e + \sum_{i=1}^{t}\sum_{e \in E(T_i)} x_e\right) \leq 2s(C) + 1,$$

and after division by two and rounding down we obtain the desired result. □

### 4.3.2 The Knapsack Polytope

Related to the capacity constraint (4.5) we have the well known **knapsack cover** inequalities

$$\sum_{i \in S} y_i \leq |S| - 1 \qquad \forall S \subseteq N, \sum_{i \in S} d_i > Q. \tag{4.19}$$

We state the validity of (4.19) without proof.

**Theorem 7** (Wolsey [35])**.** The cover inequalities (4.19) are valid for $P_K$.

### 4.3.3 The Node-Capacitated Polytope

Fischetti et al. [16] present the **path inequalities** for the orienteering problem, i.e., the $P_E$ polytope. We will adapt the inequalities to $P_N$. Let $P$ be the set of edges traversed on a partial path. Let the node set $V(P) = \{l, \ldots, k\}$ be the sequence of nodes visited on the path represented by edge set $P$ with end nodes $l$ and $k$. Let $S = \{j \in N \setminus V(P) \mid \sum_{i \in V(P)} d_i + d_j \leq Q\}$ be the set of nodes to which the path can feasibly be extended. Hence, if $\sum_{i \in V(P)} d_i > Q$ the path is in itself infeasible and $S = \emptyset$. The path inequalities are

$$\sum_{e \in P} x_e \leq \sum_{i \in V(P) \setminus \{l,k\}} y_i + \sum_{e \in E(k:S)} x_e \qquad \forall V(P) \subseteq N. \tag{4.20}$$

The edge set in the last term depends on which end of the path is extended to $S$, i.e., it could have been $E(l : S)$ instead. The validity of (4.20) for $P_E$ is proved by Fischetti et al. [16]. We adapt the proof in the following theorem.

**Theorem 8.** The path inequalities (4.20) are valid for $P_N$.

*Proof.* The proof is given by contradiction. Assume that there exist a feasible solution $(x^*, y^*)$ to CPTP violating (4.20). Then

$$x^*_{l,l+1} + (x^*_{l+1,l+2} - y^*_{l+1}) + \cdots + (x^*_{k-1,k} - y^*_{k-1}) - \sum_{j \in S} x^*_{k,j} \geq 1,$$

where $x^*_{i,i+1} - y_i \leq 0$ for all $i \in V(P) \setminus \{l, k\}$. It then follows that $x^*_{l,l+1} = 1$ (hence $y_{l+1} = 1$), $x^*_{l+1,l+2} - y_{l+1} = 0$ (hence $x^*_{l+2,l+3} = 1$ and $y_{l+3} = 1), \ldots, x^*_{k-1,k} - y^*_{k-1} = 0$, (hence $x^*_{k-1,k} = 1$), and $x^*_{k,j} = 0$ for all $j \in S$. But then the solution $(x^*, y^*)$ cannot be feasible since it contains all the edges on the path plus an edge not in $E(k : S)$. □

Bauer et al. [5] present similar inequalities for the cardinality constrained circuit problem for which the path $P$ must be infeasible.

### Multistar Inequalities

Multistar inequalities covers a family of inequalities that are related to intersection of the $P_K$ and $P_N$ polytopes. Consider the capacity inequalities given as

$$\sum_{e \in \delta(S)} x_e \geq \frac{2}{Q} \sum_{i \in S} d_i y_i \qquad \forall S \subseteq N, |S| \geq 2. \tag{4.21}$$

the inequalities ensure that any set $S$ of nodes are visited according to the resource consumption within the set $S$. The inequalities (4.21) are very similar to the fractional capacity inequalities of the CVRP (see e.g., Toth and Vigo [34]) except for the important fact that the right-hand side of (4.21) consists of variables and not constants. The capacity inequalities (4.21) can be improved by observing that nodes connected to $S$ are also visited when the connecting edge is used. Hence, the demand of those nodes can be counted on the right-hand-side. This results in what Letchford and Salazar-Gonzalez [24] denotes the *one-vehicle* generalized large multistar (GLM) inequalities

$$\sum_{e \in \delta(S)} x_e \geq \frac{2}{Q} \left( \sum_{i \in S} d_i y_i + \sum_{j \in N \setminus S} \sum_{e \in E(j:S)} d_j x_e \right) \quad \forall S \subseteq N, |S| \geq 2. \tag{4.22}$$

The GLM inequalities are generalized further by Letchford and Salazar-Gonzalez [24]. Let $a, b \geq 0$ and let the inequality $\sum_{i \in N} a_i y_i \leq b$ be valid for $P_K$. Then the *one-vehicle* knapsack large multistar (KLM) inequalities

$$\sum_{e \in \delta(S)} x_e \geq \frac{2}{b} \left( \sum_{i \in S} a_i y_i + \sum_{j \in N \setminus S} \sum_{e \in E(j:S)} a_j x_e \right) \quad \forall S \subseteq N, |S| \geq 2 \tag{4.23}$$

can be constructed. Note, that (4.21) and (4.22) correspond to (4.23) by setting $a_i = d_i$ for all $i \in S$ and $a_i = 0$ for all $i \in N \setminus S$ for (4.21) and setting $a_i = d_i$ for all $i \in C$ for (4.22) and setting $b = Q$ in both cases.

Letchford and Salazar-Gonzalez [24] state that the KLM inequalities (4.23) (and therefore (4.21) and (4.22)) can easily be proved to be valid for the CPTP (and thereby $P_N$). For completeness we bring a formal proof here:

**Theorem 9.** The KLM inequalities (4.23) are valid for $P_N$.

*Proof.* We consider three cases:

(i) Assume that $\sum_{i \in S} y_i = 0$. This implies that $\sum_{e \in E(j:S)} x_e = 0$ for all $j \in N \setminus S$ and that $\sum_{e \in \delta(S)} x_e = 0$. Therefore both the left-hand side and the right-hand side evaluates to 0.

(ii) The case where nodes in $N \setminus S$ are connected to $S$ by at most one edge. Assume $\sum_{e \in E(j:S)} x_e \leq 1$ for all $j \in N \setminus S$. This means that $\sum_{e \in E(j:S)} a_j x_e \leq a_j y_j$ and the sum on the right-hand side of (4.23) has the following relation with the corresponding knapsack inequality $\sum_{i \in S} a_i y_i + \sum_{j \in N \setminus S} \sum_{e \in E(j:S)} a_j x_e \leq \sum_{i \in N} a_i y_i \leq b$. This implies that the right-hand side can at most evaluate to 2, and since some nodes in $S$ are visited the left-hand side evaluates to at least 2.

(iii) The case where some nodes in $N \setminus S$ are connected to $S$ by more than one edge. Divide $N \setminus S$ into two disjoint sets $N'$ and $N''$. Assume $\sum_{e \in E(k:S)} x_e > 1$ for all $k \in N'$ and $\sum_{e \in E(j:S)} x_e \leq 1$ for all $j \in N''$. Since $\sum_{e \in E(k:S)} a_k x_e > a_k, k \in N'$ it may be that the sum on the right-hand side evaluates to a value more than $b$. However, since $a_k \leq b, k \in N'$ and $\sum_{e \in E(k:S)} x_e \leq 2, k \in N'$ we have $\sum_{i \in N''} a_i y_i + \sum_{k \in N'} \sum_{e \in E(k:S)} a_k x_e \leq |N'|b$ and it follows that the right-hand side of (4.23) can evaluate to at most $2 + 2|N'|$. In an integer it follows from the assumptions that $\sum_{e \in E(k:S)} x_e = 2, k \in N'$. This implies that no other edges are used from each node $k \in N'$ and it is solely connected to $S$. This in turn implies that $\sum_{e \in \delta(S) \setminus \cup_{k \in N'} E(k:S)} x_e \geq 2$ and therefore $\sum_{e \in \delta(S} x_e \geq 2 + 2|N'|$ which concludes the proof.

□

### Rounded Multistar Inequalities

In the following we will present some families of valid inequalities for $P_N$ that stems from dividing by 2 and rounding up the fractional right-hand side of the capacity inequalities (4.21) and the multistar inequalities (4.22) and (4.23). The rounding results in non-linear inequalities since we ceil a term containing variables. The non-linear inequalities are valid for $P_N$ since the left-hand side of (4.21), (4.22) and (4.23) is $\sum_{e \in \delta(S)} x_e \in \mathbb{Z}^+$ for any feasible solution which makes rounding up the right-hand side to the nearest integer okay. To obtain linear inequalities we make use of a result presented by Baldacci et al. [1] for the capacitated $m$-ring-star problem.

**Lemma 1** (Baldacci et al. [1])**.** Let $\alpha$, $b$ and $\gamma$ be three non-negative integer values with $\alpha > \gamma$ and $\text{mod}(\alpha, \gamma) \neq 0$. Then,

$$\left\lceil \frac{\alpha - \beta}{\gamma} \right\rceil \geq \left\lceil \frac{\alpha}{\gamma} \right\rceil - \frac{\beta}{\text{mod}(\alpha, \gamma)}. \tag{4.24}$$

It can be shown that (4.24) in Lemma 1 also holds even if the condition $\alpha > \gamma$ is not true. Lemma 1 is used to deduce linear inequalities that approximate the non-linear inequalities obtained from the rounding operation. Even though the GLM inequalities (4.22) dominate the capacity inequalities (4.21) this is not always true when rounding has occurred. Consider the non-linear rounded capacity inequalities stemming from (4.21)

$$\sum_{e\in\delta(S)} x_e \geq 2\left\lceil \frac{\sum_{i\in S} d_i y_i}{Q} \right\rceil \qquad \forall S \subseteq N. \tag{4.25}$$

With the use of Lemma 1 we obtain the *rounded capacity inequalities*

$$\sum_{e\in\delta(S)} x_e \geq 2\left\lceil \frac{\sum_{i\in S} d_i}{Q} \right\rceil - \frac{2\sum_{i\in S} d_i(1-y_i)}{\mathrm{mod}(\sum_{i\in S} d_i, Q)} \qquad \forall S \subseteq N. \tag{4.26}$$

**Theorem 10.** The rounded capacity inequalities (4.26) are valid for $P_N$.

*Proof.* We show that the right-hand side of (4.26) is less than or equal to the right-hand side of (4.25). We have $\sum_{i\in S} d_i y_i = \sum_{i\in S} d_i - \sum_{i\in S} d_i(1-y_i)$. Substitute the numerator of the fraction in the ceil expression on the right-hand side of (4.25) and apply Lemma 1 with non-negative values: $\alpha = \sum_{i\in S} d_i$, $\beta = \sum_{i\in S} d_i(1-y_i)$, and $\gamma = Q$ to conclude the proof. □

For briefness we consider only the KLM inequalities (4.23) next. The deduction of the non-linear rounded GLM inequalities (4.22) and their approximation is similar when we set $a_i = d_i$ for all $i \in N$, and $b = Q$. The non-linear rounded KLM inequalities are

$$\sum_{e\in\delta(S)} x_e \geq 2\left\lceil \frac{\sum_{i\in S} a_i y_i + \sum_{j\in N\setminus S}\sum_{e\in E(j:S)} a_j x_e}{b} \right\rceil \quad \forall S \subseteq N \tag{4.27}$$

and the rounded KLM (RKLM) inequalities are

$$\begin{aligned}\sum_{e\in\delta(S)} x_e \geq & 2\left\lceil \frac{\sum_{i\in S} a_i + 2\sum_{j\in N\setminus S} a_j}{b} \right\rceil \\ & - \frac{2\sum_{i\in S} a_i(1-y_i) + 2\sum_{j\in N\setminus S} a_j\left(2 - \sum_{e\in E(j:S)} x_e\right)}{\mathrm{mod}(\sum_{i\in S} a_i + 2\sum_{j\in N\setminus S} a_j, b)} \quad \forall S \subseteq N.\end{aligned} \tag{4.28}$$

**Theorem 11.** The rounded KLM inequalities (4.28) are valid for $P_N$.

*Proof.* We show that the right-hand side of (4.28) is less than or equal to the right-hand side of (4.27). We have $\sum_{i\in S} a_i y_i = \sum_{i\in S} a_i - \sum_{i\in S} a_i(1-y_i)$ and $\sum_{j\in N\setminus S}\sum_{e\in E(j:S)} a_j x_e = 2\sum_{j\in N\setminus S} a_j - \sum_{j\in N\setminus S} a_j\left(2 - \sum_{e\in E(j:S)} x_e\right)$.

Substitute the two terms in the numerator of the fraction in the ceil expression on the right-hand side of (4.27) and apply Lemma 1 with non-negative values:

$$\alpha = \sum_{i \in S} a_i + 2 \sum_{j \in N \setminus S} a_j, \beta = \sum_{i \in S} a_i (1 - y_i) + \sum_{j \in N \setminus S} a_j \left( 2 - \sum_{e \in E(j:S)} x_e \right), \gamma = b$$

to conclude the proof. □

The non-linear rounded capacity inequalities are dominated by the non-linear rounded GLM inequalities as was also the case for the original versions. However, this relation does not hold for the rounded inequalities. This is due to different values of $\alpha$ and $\beta$ in Lemma 1 used to calculate the approximation on the right-hand sides. This observation can also be transferred to the KLM inequalities. Let $\sum_{i \in N} a_i y_i \leq b$ be the valid inequality for $P_K$ on which to construct a rounded KLM inequality then for the relaxed inequality $\sum_{i \in S} a_i y_i \leq b$ we can construct a corresponding rounded capacity inequality that is not necessarily dominated.

## 4.4 Separation Results

This section discusses the separation of the inequalities used in the branch-and-cut algorithm. Particularly we discuss the running time complexity of the separation algorithms and provide heuristic procedures where appropriate.

### 4.4.1 The Circuit Polytope

The **sub-tour elimination constraints** (4.4) are separable in polynomial time by solving a series of minimum $(s, t)$-cut problems. An effective algorithm can be implemented using a Gomory-Hu cut tree. Using Goldberg-Tarjan's preflow push-relabel algorithm for solving maxflow problems, the cut tree can be calculated in $O(|V|^3 \sqrt{|E|})$ time.

#### Logical Constraints

There are $|E||N|$ logical constraints (4.10) and separation by enumeration can be done in $O(|N||E|)$ time.

#### Cut Inequalities

The separation of the cut inequalities (4.11) is similar to the separation of the sub-tour elimination constraints (4.4) although on a slightly different cut graph. Again the running time is $O(|V|^3 \sqrt{|E|})$ using a Gomory-Hu cut tree.

#### 2-Matching Inequalities

Exact separation of the 2-matching inequalities (4.13) can be done in polynomial time by solving an odd minimum cut problem, see Padberg and Rao [26]. Following in the lines of Bauer et al. [5] we have adapted a heuristic procedure presented by Padberg and Rinaldi [27]. The heuristic is Procedure 4.10 in [27] having a worst cast running time $O(|V|^4)$. As noted in [5] the running times are in practice much faster.

#### Comb Inequalities

The complexity of an exact separation algorithm for the comb inequalities (4.16) is unknown. Again we follow in line of Bauer et al. [5] and adapt a heuristic procedure for the separation of simple comb inequalities presented as Procedure 5.3 in [27].

#### Clique Tree Inequalities

As is the case for the comb inequalities, it is also unknown if there exists a polynomial time algorithm to separate the more general clique tree inequalities (4.18). Padberg and Rinaldi [27] provide a number of heuristic procedures to separate the equivalent inequalities for the TSP. However, based on our computational experience, the support graph was rarely large and dense enough to contain a violated clique tree inequality. Based on this observation we have disregarded the separation of (4.18) in this study.

### 4.4.2 The Knapsack Polytope

#### Cover Inequalities

The cover inequalities (4.19) can be separated exactly by solving a 0-1 knapsack problem which is weakly $\mathcal{NP}$-hard [35]. Separation algorithms for cover inequalities are standard in most modern mixed-integer programming solvers, so we simply use the build-in procedure for separation.

### 4.4.3 The Node-Capacitated Polytope

#### Path Inequalities

The separation problem for the path inequalities (4.20) is expected to be $\mathcal{NP}$-hard. We use an enumeration procedure as described by Fischetti et al. [16].

#### Multistar Inequalities

Letchford and Salazar-Gonzalez [24] show that the GLM inequalities (4.22) (and therefore also the capacity inequalities (4.21)) are separable in polyno-

mial time. This is done by solving a minimum $(s,t)$-cut for each node $s \in N$ and $t = 0$ on a capacitated graph where capacities depend on node $s$. The overall running time is $O(|V|^3\sqrt{|E|})$.

The exact separation of the KLM inequalities (4.23) is expected to be $\mathcal{NP}$-hard. For the CVRP, Letchford et al. [23] suggest to first find a violated cover inequality and then, using the same polynomial time algorithm as for the GLM inequalities, separate the most violated KLM inequality based on that cover. However, their results were unpromising and they resorted to only consider GLM inequalities. We have devised a new heuristic procedure to separate the KLM inequalities. The procedure is based on the following three observations:

- If we are to minimize the violation of a KLM inequality, i.e.,

$$\min \sum_{e\in\delta(S)} x_e - \frac{2}{b}\left(\sum_{i\in S} a_i y_i + \sum_{j\in N\setminus S}\sum_{e\in E(j:S)} a_j x_e\right),$$

  then to minimize the positive term $\sum_{e\in\delta(S)} x_e$ the set $S \subseteq N$ must be connected.
- For any set $S'$ that is a cover for $P_K$ there must exist some subset $S \subseteq S'$ that it is a minimal cover.
- Once a minimal cover $S$ has been determined it is possible to strengthen the inequality by lifting the inequality in $P_K$.

The first two observations leads us to a relaxed version of the separation problem for the KLM inequality. In this version we no longer seek the full KLM inequality but rather seek an inequality where $S$ is a connected cover. The heuristic used to determine such a set or more precisely a number of sets with such a property, does not take the violation into account. The heuristic is greatly inspired by the labeling algorithms for the ESPPRC but it also has a greedy aspect. The algorithm uses the concept of a label and to each label $L$ we associate the following functions:

$v(L)$ is the node the label was last extended to

$d(L)$ returns the accumulated demand

$S(L)$ returns the set of nodes the label consist of

Algorithm 1 gives the pseudo code for the first part of the heuristic, which construct the initial sets. The input for the algorithm is the original graph $G$, the capacity $Q$, an array with demands $d$, the current LP solutions $(\bar{x}, \bar{y})$, and a parameter $\tau$ that determines the minimum size of the co-boundary of the candidate sets. The value of $\tau$ influences the number of candidate sets found, and therefore also the computation time of the heuristic. The heuristic returns a candidate set $SOL$ of labels that represents connected

**Algorithm 1** The heuristic procedure to construct initial sets

1: **CONSTRUCTSETS**$(G, Q, d, \bar{x}, \bar{y}, \tau)$
2: $L \leftarrow \{0, 0, \emptyset\}$
3: $SOL \leftarrow \emptyset$
4: **for** $v \in V$ **do**
5:  **if** $v \neq 0 \wedge y_v \neq 0$ **then**
6:  $PQ.ENQUEUE(EXTENDLABEL(L, v, d))$
7:  **end if**
8: **end for**
9: **while** $PQ.TOP() \neq \emptyset$ **do**
10:  $L \leftarrow PQ.DEQUEUE()$
11:  **for** $e(v(L), v) \in \delta(v(L))$ **do**
12:  **if** $v \neq\in S(L) \wedge \bar{x}_e \neq 0 \wedge v \neq 0$ **then**
13:  $NewL \leftarrow EXTENDLABEL(L, v, d)$
14:  **if** $d(NewL) > Q$ **then**
15:  $SOL \leftarrow \{SOL \cup NewL\}$
16:  **end if**
17:  **if** $\bar{x}(\delta(S(NewL)) < \tau)$ **then**
18:  $PQ.ENQUEUE(NewL)$
19:  **end if**
20:  **end if**
21:  **end for**
22: **end while**
23: **return** $SOL$

covers. The auxiliary method $EXTENDLABEL(L, v, d)$ is used to update the data of a label that is $v(L) = v, d(L) = d(L)+d_v$ and $S(L) = S(L)\cup\{v\}$.

Line 1 to 9 initializes the priority queue $PQ$ with a label from each node that is used in the current LP solution. In line 10-11 a new label is dequeued if any exists. The for-loop in line 12 considers each node adjacent to the labels current node. In line 13 to 16 a new label is constructed if the node has not been included previously. Line 17 to 19 stores the label if it exceeds the capacity, line 20 to 24 either removes the label based on $\tau$ or enqueues it. The running time of the algorithm is in the worst case exponential, but on a sparse graph and with a high value $\tau$ it is expected to be fast.

After the first phase of the separation heuristic each label $L$ in $SOL$ represents the following valid inequality for $P_k$:

$$\sum_{i \in S(L)} y_i \leq |S(L)| - 1.$$

It should be noted that labels with identical node sets $S(L)$ may exist in which case we only consider one the node set once. Rather than converting to a KLM inequality based on the above inequality a second phase is performed where we perform sequential lifting of the variables for the nodes in $v \in V \setminus S(L)$. The procedure is as follows

- For each node $j \in V \setminus S(L)$ determine coefficient $a_j$ using the standard lifting procedure for a cover inequality (see [35]).
- Select the node $j$ where $a_j\bar{x}(S(L) : j)$ is maximal and add it to $S(L)$.
- Repeat the procedure until $a_j = 0,\ \forall j \in V \setminus S(L)$.

Based on the final set $S(L)$ and the coefficients found we convert the valid cover inequality to a KLM inequality.

#### Rounded Inequalities

The rounded capacity inequalities and the rounded GLM are not separated directly. Instead we construct these cuts based on sets identified by other separation algorithms. This include sets found in the multistar separation procedures and includes the separation of the capacity inequalities and the GLM inequalities. Both violating and non-violating cuts from the original separation algorithm is used.

The rounded KLM inequalities (and their capacity equivalent) are calculated in a similar way based on a candidate set of KLM inequalities.

## 4.5 Computational Results

The computational experiments are divided in two parts: (i) a running time comparison with dynamic programming algorithms and (ii) a detailed description of the cut separation.

All experiments are performed on a dual 2.66 GHz Intel® Xeon® X5355 machine with 16 GB of memory, no multi-threading utilized. The BAC algorithm is implemented within CPLEX 12 using the callback functions for the cut generation from the callable library. The tests are performed using the default CPLEX parameters. This includes the generation of cuts for general mixed-integer programs such as Chvátal-Gomory cuts, mixed-integer rounding cuts, disjunctive cuts, and cover inequalities.

Branching is performed by CPLEX but we enforce strong branching to reduce the search tree size. Cut separation is performed aggressively throughout the entire algorithm. Most of the cuts are separated in each iteration and one of each cut type is added if the violation is above 0.1. For the 2-matching inequalities and the comb inequalities we also enforce that the LP-induced graph is connected since the separation heuristics assume this. The KLM inequalities are only separated when no other violated inequality of another type can be found.

A time limit of 3 hours is used of for all experiments.

### 4.5.1 Benchmark Instances

We consider three sets of instances, one known from the literature and two newly generated sets.

The first set (Set 1) is used by Feillet et al. [14], Righini and Salani [29, 30] and is derived from the benchmarks by Solomon [32] for the vehicle routing problem with time windows by discarding the time constraints. All the benchmarks have 100 nodes and a depot. From the instances *c101*, *r101*, and *rc101* with three different node distributions (*c* for clustered, *r* for random, and *rc* for random-clustered) ten instances of the CPTP have been created for each distribution, where the capacity range from 10 to 100 in steps of 10. The instances are named according to the series and a tenth of the capacity, e.g., *c_101_09* is based on the *c101* instance with capacity 90. The instances with capacity from 10 to 50 are easily solved [30] so we will only consider the instances with larger capacity ranging from 60 to 100.

The second set (Set 2) of test instances is based on the VRPTW instances proposed by Gehring and Homberger [17]. We use the edge weights and demands but disregard the time windows from these instances which yields a total of 30 different instances. That is, 6 instances each with 200, 400, 600, 800 and 1000 nodes. The distribution of the nodes of the instances are identical to the benchmark instances derived from the *Solomon* instances. The difference between the type 1 and type 2 is that the latter has larger vehicles. The profits $p_i, \forall\ i \in N$, were randomly generated as an integer between 0 and 1000.

The third set (Set 3) was kindly generated by Roberto Roberti using the column generation algorithm for the CVRP presented by [3], i.e., the duals of the master problems was used to generate instances of the CPTP. The

| Set | Generated by | Derived from | # instances | # nodes |
|---|---|---|---|---|
| 1 | Feillet et al. [14] | Solomon [32] | 30 | 100 |
| 2 | Jepsen et al. | Gehring and Homberger [17] | 30 | 200-1000 |
| 3 | Roberti | E, F, M, and P series for the CVRP | 31 | 45-200 |

Table 4.1: Benchmark summary.

column generation algorithm ran without adding any additional cuts in the column generation master problem thereby ensuring that the generated sub-problems are indeed instances of the CPTP. The set contains two series of benchmarks ($a$ and $b$) based on the well-known benchmarks for the CVRP (the E, F, M, and P series available at `http://www.branchandcut.org`). The $a$ series are sub-problems generated in the "middle" of the column generation algorithm and therefore have quite negative upper bounds, and the $b$ series are sub-problems generated near the end of the column generation algorithm where the upper bound is closer to 0. For all benchmarks a valid upper bound is 0 due to the nature of the column generation algorithm (except for F-n72-k4_a where it is 0.006 due to rounding of the duals). In the hope of generating instances with a high degree of difficulty only sub-problems of a set of the more difficult instances of the CVRP have been considered. There are 17 instances in the $a$ series and 14 in the $b$ series. The problem instances differ slightly from the formulation of CPTP since a lower bound on the load is imposed based on the original CVRP instance, i.e., the minimum load is

$$d_{min} = \max \left\{ \min_{i \in V} \{d_i\}, \sum_{i \in V} d_i - (K-1)Q \right\}$$

where $K$ is the number of vehicles in the CVRP instance. This is easily modelled in the MIP model by setting a lower bound on the capacity inequality (4.5). However, a lower bound on the capacity demands special attention in the dominance rules for the dynamic programming algorithms [3].

Table 4.1 summarizes the instance sets used in the experiments.

### 4.5.2 Comparison with Dynamic Programming Algorithms

Matteo Salani has kindly provided his code for the exact dynamic programming and the decremental state space relaxation algorithms used in [29, 30]. We have implemented an updated version of the algorithms using a bounding function as suggested by Baldacci et al. [3]. Appendix A.1.1 shows an extensive computational comparison with the algorithms provided by Matteo Salani and our updated version. The experiments show that our updated implementations of the dynamic programming algorithms are superior to the

| Instance | BB Nodes | Root LB | Best LB | UB | BAC Time | DP Time |
|---|---|---|---|---|---|---|
| c_100_06 | 6 | -33 | -32 | -32 | 4.75 | 0.02 |
| c_100_07 | 0 | -41 | -41 | -41 | 2.68 | 0.05 |
| c_100_08 | 0 | -49 | -48 | -48 | 2.18 | 0.06 |
| c_100_09 | 0 | -57 | -57 | -57 | 2.42 | 0.11 |
| c_100_10 | 0 | -64 | -64 | -64 | 1.71 | 0.55 |
| r_100_06 | 6 | -62 | -61 | -61 | 2.90 | 2.74 |
| r_100_07 | 15 | -70 | -67 | -67 | 2.67 | 9.12 |
| r_100_08 | 6 | -78 | -77 | -77 | 2.65 | 14.77 |
| r_100_09 | 0 | -86 | -86 | -86 | 2.39 | 29.82 |
| r_100_10 | 0 | -93 | -93 | -93 | 2.13 | 395.64 |
| rc_100_06 | 4 | -37 | -33 | -33 | 4.21 | 0.01 |
| rc_100_07 | 20 | -42 | -34 | -34 | 3.87 | 0.02 |
| rc_100_08 | 8 | -46 | -42 | -42 | 2.43 | 0.03 |
| rc_100_09 | 10 | -52 | -51 | -51 | 4.58 | 0.04 |
| rc_100_10 | 32 | -57 | -53 | -53 | 4.58 | 0.11 |

Table 4.2: Comparison of the BAC algorithm with the fastest dynamic programming algorithm (see Table A.8 in Appendix A.1.1) for Set 1.

algorithms provided by Matteo Salani, so for the remainder of this paper we will only compare with our own implementations.

The results of the comparison between the BAC algorithm and the fastest dynamic programming algorithm are shown in Tables 4.2, 4.3 and 4.4. The tables contain the following columns: *Instance* is the name of the instance, *BB* is the number of branch nodes, *Root LB* is the bound in the root node, *Best LB* is the best lower bound obtained, *UB* is the best solution obtained by the BAC algorithm, *BAC Time* is the time of the BAC algorithm, and *DP Time* is the time of the best performing dynamic programming algorithm (appendix A.1.1). A dash (-) in any of the time fields indicate that the algorithm was terminated after 10000 CPU seconds and an asterisk ( *) indicates that the algorithm ran out of memory.

The results for Set 1 and Set 2 in Tables 4.2 and 4.3 are very encouraging. Although the fastest dynamic programming algorithm is faster on most of the 100 node instances in Set 1, it is seen that where the dynamic programming algorithm have trouble, e.g., instances r_100_09, r_100_10, the BAC algorithm is by far superior. This trend continues for the much larger instances in Set 2 where the BAC algorithm is able to solve to optimality 18 instances, the largest of these contains 800 nodes, and the dynamic programming algorithm is not able to prove optimality on any of the instances. Several instances are solved in the root node for both the instances in Set 1 and Set 2 and for the instances which are solved to optimality no more than 130 branching nodes are used.

Table 4.4 presents the results for Set 3 which is less encouraging than the previous results. The dynamic programming algorithms outperforms

| Instance | BB Nodes | Root LB | Best LB | UB | BAC Time | DP Time |
|---|---|---|---|---|---|---|
| c1_2_a | 0 | -12.988 | -12.983 | -12.983 | 10.28 | - |
| c1_4_a | 34 | -14.128 | -13.984 | -13.984 | 77.22 | - |
| c1_6_a | 21 | -12.973 | -12.906 | -12.906 | 724.94 | - |
| c1_8_a | 23 | -13.770 | -13.555 | -13.555 | 5104.55 | - |
| c1_10_a | 0 | -13.656 | -13.656 | -12.856 | * | - |
| c2_2_a | 0 | -37.261 | -37.261 | -37.261 | 36.68 | - |
| c2_4_a | 0 | -44.375 | -44.375 | -44.375 | 1282.32 | - |
| c2_6_a | 0 | -47.829 | -47.829 | 0.000 | * | - |
| c2_8_a | 0 | -45.253 | -45.253 | -37.932 | * | - |
| c2_10_a | 0 | -46.573 | -46.573 | 0.000 | - | - |
| r1_2_a | 53 | -15.112 | -15.005 | -15.005 | 18.59 | - |
| r1_4_a | 64 | -18.266 | -18.207 | -18.207 | 110.70 | - |
| r1_6_a | 130 | -20.081 | -19.993 | -19.993 | 723.13 | - |
| r1_8_a | 0 | -18.313 | -18.313 | -14.838 | * | - |
| r1_10_a | 0 | -14.791 | -14.791 | -12.282 | * | - |
| r2_2_a | 46 | -48.568 | -48.509 | -48.509 | 214.63 | - |
| r2_4_a | 63 | -59.501 | -59.482 | -59.482 | 566.72 | - |
| r2_6_a | 104 | -69.603 | -69.562 | -69.562 | 9762.00 | - |
| r2_8_a | 0 | -71.001 | -71.001 | 0.000 | * | - |
| r2_10_a | 0 | -63.697 | -63.697 | 0.000 | * | - |
| rc1_2_a | 1 | -13.885 | -13.872 | -13.872 | 6.67 | - |
| rc1_4_a | 17 | -15.324 | -15.239 | -15.239 | 77.31 | - |
| rc1_6_a | 59 | -15.521 | -15.389 | -15.389 | 333.30 | - |
| rc1_8_a | 2 | -14.979 | -14.962 | -14.962 | 3030.01 | - |
| rc1_10_a | 0 | -14.524 | -14.524 | -2.457 | * | - |
| rc2_2_a | 65 | -48.295 | -48.246 | -48.246 | 198.83 | - |
| rc2_4_a | 5 | -57.952 | -57.947 | -57.947 | 3035.06 | - |
| rc2_6_a | 0 | -63.691 | -63.691 | -49.937 | * | - |
| rc2_8_a | 0 | -63.603 | -63.603 | 0.000 | - | - |
| rc2_10_a | 0 | -69.129 | -69.129 | 0.000 | * | - |

Table 4.3: Comparison of the BAC algorithm with the fastest dynamic programming algorithm (see Table A.9 in Appendix A.1.1) for Set 2

| Instance | BB Nodes | Root LB | Best LB | UB | BAC Time | DP Time |
|---|---|---|---|---|---|---|
| E-n76-k7_a | 93 | -28.645 | -6.032 | -6.032 | 43.08 | 0.10 |
| E-n76-k7_b | 269 | -23.801 | -0.846 | -0.846 | 83.17 | 0.05 |
| E-n76-k8_a | 107 | -30.685 | -6.635 | -6.635 | 52.01 | 0.24 |
| E-n76-k8_b | 616 | -27.997 | -0.001 | -0.001 | 185.19 | 0.28 |
| E-n76-k10_a | 325 | -34.486 | -3.810 | -3.810 | 162.50 | 0.15 |
| E-n76-k10_b | 795 | -32.380 | 0.000 | 0.000 | 258.27 | 0.29 |
| E-n76-k14_a | 417 | -37.667 | -3.788 | -3.788 | 136.41 | 0.12 |
| E-n76-k14_b | 1028 | -36.150 | -0.002 | -0.002 | 308.18 | 0.12 |
| E-n101-k8_a | 12 | -29.846 | -23.977 | -23.977 | 20.65 | 5.06 |
| E-n101-k8_b | 974 | -22.491 | -0.006 | -0.006 | 498.92 | 0.14 |
| E-n101-k14_a | 136 | -37.179 | -6.667 | -6.667 | 258.31 | 0.04 |
| E-n101-k14_b | 959 | -33.830 | -0.002 | -0.002 | 406.73 | 0.03 |
| F-n45-k4_a | 57 | -17.272 | -1.001 | -1.001 | 2.27 | 1.27 |
| F-n72-k4_a | 22 | -2.563 | 0.006 | 0.006 | 5.38 | 102.06 |
| F-n135-k7_a | 62 | -92.869 | -9.051 | -9.051 | 653.87 | - |
| M-n121-k7_a | 69 | -81.097 | -15.260 | -15.260 | 939.83 | - |
| M-n121-k7_b | 204 | -81.733 | -2.179 | -2.179 | 1843.44 | - |
| M-n151-k12_a | 186 | -34.198 | -5.799 | -5.799 | 1345.53 | 0.77 |
| M-n151-k12_b | 1818 | -32.153 | -0.002 | -0.002 | 5841.19 | 0.32 |
| M-n200-k16_a | 175 | -43.288 | -11.136 | 0.000 | - | 36.05 |
| M-n200-k16_b | 157 | -42.113 | -10.527 | 0.000 | - | 20.84 |
| M-n200-k17_a | 383 | -44.149 | -7.328 | -5.102 | - | 0.09 |
| M-n200-k17_b | 475 | -36.907 | -2.752 | 0.000 | - | 2.25 |
| P-n70-k10_a | 425 | -29.562 | -2.852 | -2.852 | 103.22 | 0.11 |
| P-n70-k10_b | 731 | -26.953 | -0.001 | -0.001 | 150.00 | 0.10 |
| P-n76-k4_a | 93 | -12.588 | -2.903 | -2.903 | 35.35 | 1.67 |
| P-n76-k4_b | 436 | -11.480 | 0.000 | 0.000 | 84.94 | 1.39 |
| P-n76-k5_a | 198 | -18.549 | -3.959 | -3.959 | 92.04 | 0.58 |
| P-n101-k4_a | 72 | -13.181 | -7.219 | -7.219 | 27.93 | 3455.28 |
| P-n101-k4_b | 239 | -11.974 | -0.001 | -0.001 | 106.39 | - |

Table 4.4: Comparison of the BAC algorithm with the fastest dynamic programming algorithm (see Table A.10 in Appendix A.1.1) for Set 3

the BAC algorithm on most of the instances solved by both algorithms and is able to solve 4 instances not solved by the BAC algorithm. Especially the *b* instances in this set are difficult for the BAC algorithm and it is not able to solve the four large instances with 200 nodes that are solved in less than 40 seconds by the dynamic programming algorithms. One reason why the dynamic programming algorithms perform so well on these instances is that the path relaxations gives fairly good bounds resulting in large number of labels being fathomed and few states being explored. This thesis is supported by the results in Table A.10 in Appendix A.1.1. On the positive side, the BAC algorithm is able to solve 3 instances that are not solved by the dynamic programming algorithms and it is on two instances faster. The root lower bound is significantly weaker compared to the instances in Set 1 and Set 2 and the number of branching nodes is much higher than in the other two sets.

### 4.5.3 Cut Separation

Details about cut separation are presented in Tables 4.5, 4.6 and 4.7. Each table contains the following columns: *Instance* is the name of the instance, and for each type of cut we report the number of cuts separated $N$ of the inequalities and the time $T$ used in the separation algorithm. The columns are named for each cut as: *GSEC* for the GSEC constraints (4.4), *GLM* for the GLM inequalities (4.22), *RGLM* for the rounded GLM inequalities (4.28) with $a_i = d_i, \forall i \in V$ and $b = Q$, *CAP* for the capacity inequalities (4.21), *RCAP* for the rounded capacity inequalities (4.26), *KLM* for the KLM inequalities (4.23), *RKLM* for the rounded KLM inequalities (4.28), *PATH* for the path inequalities (4.20), *TWOM* for the 2-matching inequalities (4.13), and *COMB* for the comb inequalities (4.16). The last two columns are: *TT* reports the total running time of the BAC algorithm and column *TC* reports the total time spent in all the separation algorithms. For the rounded inequalities no separation time is reported because it is negligible. This is due to the fact that the candidate set for the rounded inequalities are found when separating the non-rounded version.

From the detailed cut statistics in Tables 4.5, 4.6, and 4.7 we can see that the GSEC inequalities are the most frequently separated cut. The GLM inequalities are the second most separated cut with a few exceptions in Set 3. The rounded GLM inequalities and the rounded CAP inequalities seems to be the third and fourth most generated cuts. They are in some cases separated almost as often as the GLM inequalities, but in other cases the GLM inequalities are separated more than six times as often. The KLM inequalities appear to be fifth most generated cut, especially on the *b* instances in Set 2 it is frequently generated and in two cases it is the second most separated cut.

There are several instances where we find no path inequalities. This indicates, that for the BAC algorithm the path structure is seldom kept intact enough in fractional solutions for path inequalities to be violated. The same seem to be the case for the 2-matching and the comb inequalities, which both are found quite seldom.

We are unable to separate any violated capacity or rounded KLM inequalities. For the capacity inequalities this is probably due to the fact that the GLM inequalities are separated earlier in the process and therefore none of the CAP inequalities are found in later iterations. For the rounded KLM inequalities we believe that the issue is that few KLM inequalities are found and that the constants that are used to ensure that the rounded inequalities are valid are too large, thereby not leading to any violation.

In Set 2 the GSEC inequalities are the most time consuming to separate. However, as the instances grow in size the GLM inequalities become more time consuming to separate and especially on the larger *c2*, *r2* and *rc2* instances a significant amount of time is spent on separation. However, the

overall separation time used is seldom above 6 % of the total CPU time and only in a single case the separation time exceeds 10 %, which is due to a very high separation time in the separation algorithm for the 2-matching inequalities.

## 4.6 Conclusion

This paper presented a BAC algorithm for the undirected CPTP. We have reviewed the existing literature in the area of tour problems and have pointed out that many of the known problems are related (or is in fact identical) to the CPTP. We have adapted a number of valid inequalities for related problems to the CPTP and proved the validity of these inequalities. Moreover, we have presented a new family of inequalities, the rounded multistar inequalities, for the CPTP.

Separation algorithms have been implemented for all cuts and the usefulness of the valid inequalities has been shown through a computational study. Especially the GLM, the rounded GLM inequalities, and the rounded capacity inequalities proved to be frequently separated. Furthermore, a comparison with state-of-the-art dynamic programming algorithms has shown that the BAC algorithm is competitive, and acts as a good compliment to the dynamic programming algorithms. That is, in some case the dynamic programming algorithms are much faster and able to solve instances that cannot be solved by the BAC algorithm. On the other hand the BAC algorithm appear to superior on very large instances, e.g., the BAC algorithm solved instances with up to 800 nodes compared to a maximum of 200 nodes for the dynamic programming algorithms. There is a tendency towards, that the BAC algorithm is faster than the dynamic programming algorithms on instances with highly negative weights and is slower on instances with solution values close to 0. This is not too surprising since the bounding functions used in the dynamic programming algorithms is expected to cut off large parts state-space in the latter case. This may render the BAC algorithms less efficient in a column generation scheme for current state-of-the-art algorithms, since instances with much negativity are usually solved heuristically and only the instances with cost near zero are solved to optimality. However, our experiments indicate that the BAC algorithm may prove to be worthwhile when the number of nodes increases (to more than the current maximum of 151 nodes for the CVRP). To sum up, the BAC algorithm solved 58 out of a total 76 instances which is 18 more than the dynamic programming algorithms, and the BAC algorithm appear to have its strength when the number of nodes is large.

## Acknowledgements

Thanks to Matteo Salani for providing the code for the labeling algorithm used in the papers by Righini and Salani [29, 30]. Thanks to Roberto Roberti for providing the benchmark instances in Set 3. Thanks to Adam Letchford for pointing out the relation between the *one-vehicle* KLM inequalities and a family of valid inequalities presented in a preliminary version of this paper.

| Instance | GSEC | | GLM | | RGLM | CAP | | RCAP | KLM | | RKLM | PATH | | TWOM | | COMB | | TC | | TT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | N | T | N | T | N | N | T | N | N | T | N | N | T | N | T | N | T | N | T | |
| c_100_06 | 50 | 0.05 | 16 | 0.00 | 11 | 0 | 0.00 | 11 | 3 | 0.02 | 0 | 2 | 0.00 | 0 | 0.01 | 0 | 0.01 | 93 | 0.09 | 4.75 |
| c_100_07 | 44 | 0.00 | 21 | 0.00 | 16 | 0 | 0.00 | 16 | 0 | 0.00 | 0 | 2 | 0.00 | 0 | 0.00 | 0 | 0.00 | 99 | 0.00 | 2.68 |
| c_100_08 | 40 | 0.00 | 28 | 0.00 | 20 | 0 | 0.00 | 20 | 1 | 0.00 | 0 | 1 | 0.01 | 0 | 0.00 | 0 | 0.01 | 110 | 0.02 | 2.18 |
| c_100_09 | 52 | 0.01 | 32 | 0.01 | 28 | 0 | 0.00 | 29 | 0 | 0.00 | 0 | 0 | 0.01 | 0 | 0.00 | 0 | 0.00 | 141 | 0.03 | 2.42 |
| c_100_10 | 42 | 0.01 | 29 | 0.03 | 23 | 0 | 0.00 | 23 | 0 | 0.00 | 0 | 0 | 0.00 | 0 | 0.00 | 0 | 0.00 | 117 | 0.04 | 1.71 |
| r_100_06 | 67 | 0.05 | 55 | 0.07 | 18 | 0 | 0.00 | 17 | 1 | 0.01 | 0 | 2 | 0.03 | 0 | 0.00 | 0 | 0.00 | 160 | 0.16 | 2.90 |
| r_100_07 | 88 | 0.10 | 66 | 0.05 | 19 | 0 | 0.02 | 20 | 2 | 0.05 | 0 | 0 | 0.03 | 1 | 0.02 | 1 | 0.02 | 197 | 0.29 | 2.67 |
| r_100_08 | 67 | 0.05 | 61 | 0.06 | 21 | 0 | 0.04 | 22 | 0 | 0.05 | 0 | 2 | 0.04 | 0 | 0.00 | 0 | 0.04 | 173 | 0.28 | 2.65 |
| r_100_09 | 76 | 0.08 | 66 | 0.11 | 26 | 0 | 0.01 | 27 | 1 | 0.02 | 0 | 0 | 0.01 | 0 | 0.03 | 0 | 0.03 | 196 | 0.29 | 2.39 |
| r_100_10 | 74 | 0.10 | 68 | 0.09 | 30 | 0 | 0.00 | 28 | 0 | 0.00 | 0 | 0 | 0.03 | 0 | 0.01 | 0 | 0.01 | 200 | 0.24 | 2.13 |
| rc_100_06 | 79 | 0.04 | 23 | 0.03 | 20 | 0 | 0.00 | 20 | 1 | 0.02 | 0 | 21 | 0.01 | 0 | 0.01 | 1 | 0.00 | 165 | 0.11 | 4.21 |
| rc_100_07 | 124 | 0.21 | 24 | 0.01 | 16 | 0 | 0.03 | 16 | 26 | 0.23 | 0 | 43 | 0.07 | 0 | 0.07 | 4 | 0.06 | 253 | 0.68 | 3.87 |
| rc_100_08 | 61 | 0.06 | 29 | 0.02 | 12 | 0 | 0.02 | 12 | 5 | 0.08 | 0 | 13 | 0.00 | 0 | 0.02 | 0 | 0.00 | 132 | 0.20 | 2.43 |
| rc_100_09 | 179 | 0.56 | 29 | 0.09 | 22 | 0 | 0.00 | 22 | 5 | 0.15 | 0 | 33 | 0.06 | 0 | 0.01 | 0 | 0.05 | 290 | 0.92 | 4.58 |
| rc_100_10 | 103 | 0.11 | 39 | 0.05 | 18 | 0 | 0.04 | 17 | 7 | 0.17 | 0 | 13 | 0.05 | 0 | 0.04 | 0 | 0.09 | 197 | 0.55 | 4.58 |

Table 4.5: A detailed result of the frequency of cuts separated on Set 1.

| Instance | GSEC | | GLM | | RGLM | CAP | | RCAP | KLM | | RKLM | PATH | | TWOM | | COMB | | TC | | TT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | N | T | N | T | N | N | T | N | N | T | N | N | T | N | T | N | T | N | T | |
| c1_2_a | 50 | 0.03 | 41 | 0.12 | 32 | 0 | 0.02 | 29 | 0 | 0.03 | 0 | 0 | 0.07 | 0 | 0.02 | 0 | 0.03 | 152 | 0.32 | 10.28 |
| c1_4_a | 209 | 0.68 | 85 | 1.07 | 51 | 0 | 0.19 | 53 | 0 | 0.58 | 0 | 0 | 0.58 | 3 | 5.50 | 2 | 0.77 | 403 | 9.37 | 77.22 |
| c1_6_a | 1104 | 20.78 | 807 | 10.48 | 117 | 0 | 0.40 | 126 | 0 | 3.53 | 0 | 1 | 6.20 | 0 | 1.23 | 0 | 2.45 | 2155 | 45.07 | 724.94 |
| c1_8_a | 5229 | 161.52 | 556 | 62.17 | 92 | 0 | 0.86 | 93 | 0 | 6.13 | 0 | 1 | 63.08 | 0 | 3.37 | 0 | 5.45 | 5971 | 302.58 | 5104.55 |
| c1_10_a | 4335 | 146.88 | 579 | 107.34 | 116 | 0 | 0.02 | 118 | 0 | 0.00 | 0 | 0 | 76.26 | 0 | 1.47 | 0 | 1.28 | 5148 | 333.25 | 9967.54 |
| c2_2_a | 152 | 1.24 | 149 | 4.92 | 140 | 0 | 0.22 | 115 | 0 | 0.07 | 0 | 0 | 0.27 | 2 | 0.63 | 1 | 0.03 | 559 | 7.38 | 36.68 |
| c2_4_a | 1022 | 31.65 | 908 | 123.96 | 450 | 0 | 0.98 | 463 | 0 | 0.60 | 0 | 0 | 3.60 | 3 | 11.29 | 2 | 0.23 | 2848 | 172.31 | 1282.32 |
| c2_6_a | 2833 | 108.29 | 2833 | 338.60 | 1593 | 0 | 0.00 | 1605 | 0 | 0.00 | 0 | 0 | 22.28 | 0 | 0.00 | 0 | 0.00 | 8864 | 469.17 | 9541.85 |
| c2_8_a | 1802 | 82.81 | 1798 | 234.69 | 948 | 0 | 0.10 | 920 | 0 | 0.00 | 0 | 0 | 27.54 | 0 | 0.26 | 0 | 0.24 | 5468 | 345.64 | 7463.77 |
| c2_10_a | 2134 | 111.33 | 2134 | 243.14 | 1010 | 0 | 0.00 | 933 | 0 | 0.00 | 0 | 0 | 31.82 | 0 | 0.00 | 0 | 0.00 | 6211 | 386.29 | 10000.09 |
| r1_2_a | 97 | 0.28 | 80 | 0.83 | 77 | 0 | 0.35 | 68 | 5 | 0.48 | 0 | 1 | 0.26 | 8 | 3.43 | 3 | 0.29 | 339 | 5.92 | 18.59 |
| r1_4_a | 130 | 0.78 | 108 | 2.29 | 91 | 0 | 1.01 | 88 | 2 | 1.78 | 0 | 0 | 0.95 | 1 | 4.90 | 0 | 1.23 | 420 | 12.94 | 110.70 |
| r1_6_a | 161 | 1.56 | 46 | 5.91 | 59 | 0 | 3.73 | 55 | 15 | 7.63 | 0 | 0 | 3.37 | 9 | 57.65 | 13 | 8.74 | 358 | 88.59 | 723.13 |
| r1_8_a | 3072 | 87.43 | 3068 | 144.66 | 650 | 0 | 0.05 | 676 | 0 | 0.00 | 0 | 0 | 36.38 | 0 | 0.42 | 0 | 0.40 | 7466 | 269.34 | 9942.50 |
| r1_10_a | 3116 | 125.99 | 3113 | 127.23 | 379 | 0 | 0.03 | 373 | 0 | 0.00 | 0 | 0 | 61.87 | 0 | 0.57 | 0 | 0.49 | 6981 | 316.18 | 9989.84 |
| r2_2_a | 176 | 8.76 | 127 | 30.98 | 123 | 0 | 15.08 | 97 | 0 | 7.14 | 0 | 0 | 0.65 | 51 | 46.41 | 12 | 0.98 | 586 | 110.00 | 214.63 |
| r2_4_a | 178 | 7.15 | 121 | 41.28 | 86 | 0 | 20.50 | 87 | 6 | 21.75 | 0 | 0 | 1.81 | 24 | 41.54 | 12 | 1.88 | 514 | 135.91 | 566.72 |
| r2_6_a | 1530 | 147.72 | 1496 | 668.79 | 964 | 0 | 88.42 | 979 | 5 | 68.97 | 0 | 0 | 16.39 | 29 | 483.73 | 0 | 12.74 | 5003 | 1486.76 | 9762.00 |
| r2_8_a | 1264 | 84.61 | 1264 | 429.15 | 1262 | 0 | 0.00 | 1264 | 0 | 0.00 | 0 | 0 | 18.39 | 0 | 0.00 | 0 | 0.00 | 5054 | 532.15 | 9981.80 |
| r2_10_a | 1045 | 72.00 | 1045 | 317.40 | 614 | 0 | 0.00 | 625 | 0 | 0.00 | 0 | 0 | 23.07 | 0 | 0.00 | 0 | 0.00 | 3329 | 412.47 | 9954.33 |
| rc1_2_a | 50 | 0.05 | 36 | 0.18 | 22 | 0 | 0.01 | 23 | 0 | 0.01 | 0 | 0 | 0.05 | 0 | 0.07 | 0 | 0.02 | 131 | 0.39 | 6.67 |
| rc1_4_a | 240 | 1.01 | 230 | 1.03 | 99 | 0 | 0.14 | 102 | 2 | 0.44 | 0 | 0 | 0.82 | 0 | 1.19 | 0 | 0.50 | 673 | 5.13 | 77.31 |
| rc1_6_a | 335 | 1.98 | 275 | 3.86 | 146 | 0 | 1.05 | 130 | 2 | 2.37 | 0 | 2 | 2.20 | 1 | 11.97 | 2 | 3.02 | 893 | 26.45 | 333.30 |
| rc1_8_a | 1371 | 24.57 | 1361 | 22.03 | 194 | 0 | 0.13 | 211 | 0 | 0.58 | 0 | 0 | 14.79 | 1 | 1.24 | 0 | 0.84 | 3138 | 64.18 | 3030.01 |
| rc1_10_a | 3426 | 121.83 | 194 | 12.60 | 131 | 0 | 0.00 | 130 | 0 | 0.00 | 0 | 0 | 66.25 | 0 | 0.54 | 0 | 0.47 | 3881 | 201.69 | 9988.38 |
| rc2_2_a | 193 | 7.04 | 116 | 28.43 | 48 | 0 | 13.43 | 48 | 3 | 7.91 | 0 | 0 | 0.79 | 36 | 48.89 | 17 | 0.75 | 461 | 107.24 | 198.83 |
| rc2_4_a | 1518 | 52.29 | 1515 | 214.66 | 904 | 0 | 3.20 | 939 | 1 | 2.39 | 0 | 0 | 6.27 | 1 | 3.48 | 0 | 0.41 | 4878 | 282.70 | 3035.06 |
| rc2_6_a | 2373 | 135.39 | 2373 | 504.03 | 1000 | 0 | 0.10 | 1037 | 0 | 0.00 | 0 | 0 | 20.38 | 0 | 0.03 | 0 | 0.03 | 6783 | 659.96 | 9996.13 |
| rc2_8_a | 1558 | 137.90 | 1558 | 530.61 | 662 | 0 | 0.00 | 668 | 0 | 0.00 | 0 | 0 | 16.18 | 0 | 0.00 | 0 | 0.00 | 4446 | 684.69 | 10005.53 |
| rc2_10_a | 1064 | 85.01 | 1064 | 369.32 | 774 | 0 | 0.00 | 746 | 0 | 0.00 | 0 | 0 | 21.24 | 0 | 0.00 | 0 | 0.00 | 3648 | 475.57 | 9993.88 |

Table 4.6: A detailed result of the frequency of cuts separated on Set 2.

| Instance | GSEC | | GLM | | RGLM | CAP | | RCAP | KLM | | RKLM | PATH | | TWOM | | COMB | | TC | | TT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | N | T | N | T | N | N | T | N | N | T | N | N | T | N | T | N | T | N | T | |
| E-n76-k7_a | 560 | 2.54 | 188 | 0.45 | 32 | 0 | 0.08 | 26 | 9 | 0.83 | 0 | 13 | 0.24 | 0 | 0.08 | 3 | 0.27 | 831 | 4.49 | 43.08 |
| E-n76-k7_b | 661 | 3.15 | 207 | 0.57 | 24 | 0 | 0.18 | 34 | 36 | 1.53 | 0 | 12 | 0.28 | 1 | 0.25 | 1 | 0.42 | 976 | 6.38 | 83.17 |
| E-n76-k8_a | 599 | 2.07 | 240 | 0.40 | 61 | 0 | 0.08 | 65 | 15 | 0.86 | 0 | 13 | 0.16 | 0 | 0.08 | 2 | 0.27 | 995 | 3.92 | 52.01 |
| E-n76-k8_b | 967 | 3.58 | 232 | 1.13 | 70 | 0 | 0.52 | 71 | 122 | 2.03 | 0 | 13 | 0.39 | 0 | 0.38 | 2 | 0.65 | 1477 | 8.68 | 185.19 |
| E-n76-k10_a | 714 | 2.32 | 206 | 0.60 | 34 | 0 | 0.21 | 34 | 66 | 1.54 | 0 | 25 | 0.24 | 0 | 0.17 | 1 | 0.54 | 1080 | 5.62 | 162.50 |
| E-n76-k10_b | 804 | 2.99 | 231 | 0.89 | 50 | 0 | 0.33 | 55 | 217 | 2.06 | 0 | 36 | 0.30 | 0 | 0.51 | 1 | 0.77 | 1394 | 7.85 | 258.27 |
| E-n76-k14_a | 456 | 1.26 | 161 | 0.32 | 21 | 0 | 0.21 | 23 | 138 | 1.31 | 0 | 25 | 0.06 | 0 | 0.28 | 0 | 0.61 | 824 | 4.05 | 136.41 |
| E-n76-k14_b | 554 | 1.74 | 159 | 0.48 | 21 | 0 | 0.39 | 23 | 292 | 1.76 | 0 | 43 | 0.24 | 0 | 0.54 | 2 | 0.82 | 1094 | 5.97 | 308.18 |
| E-n101-k8_a | 493 | 3.45 | 90 | 0.33 | 52 | 0 | 0.00 | 52 | 1 | 0.51 | 0 | 1 | 0.15 | 0 | 0.00 | 0 | 0.09 | 689 | 4.53 | 20.65 |
| E-n101-k8_b | 1411 | 9.09 | 218 | 2.63 | 40 | 0 | 1.20 | 45 | 156 | 6.77 | 0 | 2 | 0.79 | 6 | 1.30 | 9 | 1.54 | 1887 | 23.32 | 498.92 |
| E-n101-k14_a | 963 | 4.40 | 378 | 0.79 | 39 | 0 | 0.12 | 38 | 17 | 2.10 | 0 | 8 | 0.30 | 0 | 0.14 | 0 | 0.72 | 1443 | 8.57 | 258.31 |
| E-n101-k14_b | 870 | 4.75 | 321 | 0.90 | 30 | 0 | 0.57 | 33 | 198 | 4.09 | 0 | 15 | 0.43 | 0 | 1.00 | 1 | 1.59 | 1468 | 13.33 | 406.73 |
| F-n45-k4_a | 119 | 0.14 | 30 | 0.02 | 25 | 0 | 0.05 | 25 | 4 | 0.04 | 0 | 7 | 0.02 | 0 | 0.03 | 0 | 0.00 | 210 | 0.30 | 2.27 |
| F-n72-k4_a | 160 | 0.53 | 50 | 0.11 | 25 | 0 | 0.01 | 24 | 2 | 0.10 | 0 | 12 | 0.03 | 0 | 0.00 | 0 | 0.05 | 273 | 0.83 | 5.38 |
| F-n135-k7_a | 1729 | 27.53 | 370 | 1.47 | 94 | 0 | 0.51 | 95 | 4 | 2.27 | 0 | 6 | 1.33 | 0 | 0.13 | 0 | 0.41 | 2298 | 33.65 | 653.87 |
| M-n121-k7_a | 1458 | 20.02 | 482 | 1.79 | 87 | 0 | 0.30 | 84 | 1 | 3.73 | 0 | 19 | 1.73 | 0 | 0.14 | 0 | 0.65 | 2131 | 28.36 | 939.83 |
| M-n121-k7_b | 1642 | 24.95 | 478 | 3.35 | 77 | 0 | 0.83 | 84 | 8 | 6.79 | 0 | 17 | 2.39 | 1 | 0.56 | 0 | 1.38 | 2307 | 40.25 | 1843.44 |
| M-n151-k12_a | 1816 | 26.38 | 685 | 5.30 | 112 | 0 | 0.60 | 110 | 7 | 10.03 | 0 | 4 | 1.22 | 1 | 0.48 | 1 | 2.36 | 2736 | 46.37 | 1345.53 |
| M-n151-k12_b | 2227 | 37.05 | 854 | 11.18 | 87 | 0 | 4.74 | 79 | 292 | 27.22 | 0 | 0 | 2.12 | 2 | 4.34 | 4 | 6.77 | 3545 | 93.42 | 5841.19 |
| M-n200-k16_a | 2641 | 80.54 | 1494 | 12.57 | 51 | 0 | 0.56 | 58 | 6 | 31.97 | 0 | 2 | 2.18 | 0 | 0.77 | 2 | 5.86 | 4254 | 134.45 | 10000.18 |
| M-n200-k16_b | 2923 | 81.06 | 1619 | 13.34 | 69 | 0 | 0.71 | 80 | 11 | 30.85 | 0 | 6 | 2.87 | 0 | 0.79 | 0 | 4.84 | 4708 | 134.46 | 9999.97 |
| M-n200-k17_a | 3419 | 94.78 | 1623 | 15.38 | 75 | 0 | 1.32 | 65 | 8 | 47.46 | 0 | 4 | 2.92 | 0 | 1.43 | 0 | 8.61 | 5194 | 171.90 | 10000.17 |
| M-n200-k17_b | 2833 | 81.19 | 1302 | 15.07 | 78 | 0 | 2.12 | 78 | 9 | 57.18 | 0 | 2 | 3.01 | 0 | 2.13 | 0 | 10.55 | 4302 | 171.25 | 10000.84 |
| P-n70-k10_a | 485 | 1.42 | 170 | 0.32 | 23 | 0 | 0.17 | 20 | 120 | 1.03 | 0 | 28 | 0.19 | 0 | 0.19 | 3 | 0.59 | 849 | 3.91 | 103.22 |
| P-n70-k10_b | 493 | 1.54 | 172 | 0.44 | 31 | 0 | 0.27 | 25 | 185 | 1.16 | 0 | 34 | 0.20 | 0 | 0.39 | 2 | 0.51 | 942 | 4.51 | 150.00 |
| P-n76-k4_a | 500 | 3.42 | 141 | 0.72 | 25 | 0 | 0.29 | 31 | 2 | 1.19 | 0 | 2 | 0.38 | 3 | 0.10 | 2 | 0.12 | 706 | 6.22 | 35.35 |
| P-n76-k4_b | 583 | 4.22 | 143 | 1.79 | 33 | 0 | 0.83 | 34 | 56 | 3.27 | 0 | 7 | 0.56 | 7 | 0.46 | 5 | 0.34 | 868 | 11.47 | 84.94 |
| P-n76-k5_a | 534 | 3.71 | 150 | 1.01 | 26 | 0 | 0.38 | 23 | 11 | 2.46 | 0 | 8 | 0.25 | 2 | 0.26 | 3 | 0.22 | 757 | 8.29 | 92.04 |
| P-n76-k5_b | 637 | 3.78 | 155 | 1.25 | 27 | 0 | 0.85 | 27 | 108 | 2.42 | 0 | 7 | 0.42 | 7 | 0.48 | 5 | 0.45 | 973 | 9.65 | 94.52 |
| P-n101-k4_a | 418 | 4.26 | 136 | 1.19 | 39 | 0 | 0.67 | 40 | 2 | 1.21 | 0 | 0 | 0.43 | 0 | 0.14 | 0 | 0.16 | 635 | 8.06 | 27.93 |
| P-n101-k4_b | 701 | 8.11 | 165 | 2.70 | 40 | 0 | 1.15 | 53 | 9 | 4.70 | 0 | 0 | 0.42 | 18 | 0.69 | 9 | 0.54 | 995 | 18.31 | 106.39 |

Table 4.7: A detailed result of the frequency of cuts separated on Set 3.

# Bibliography


[1] R. Baldacci, M. Dell'Amico, and J. Salazar-Gonzalez. The capacitated m-ring-star problem. *Operations Research*, 55(6):1147–1162, 2007. doi: 10.1287/opre.1070.0432.

[2] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008. doi: 10.1007/s10107-007-0178-5.

[3] R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. Working paper, 2010.

[4] P. Bauer. The circuit polytope: facets. *Mathematics of Operations Research*, 22:110–145, 1997.

[5] P. Bauer, J. T. Linderoth, and M. W. P. Savelsbergh. A branch and cut approach to the cardinality constrained circuit problem. *Mathematical Programming*, 91(2):307–348, 2002. doi: 10.1007/s101070100209.

[6] J.E. Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19:379–394, 1989. doi: 10.1002/net.3230190402.

[7] A.E. Bixby. *Polyhedral analysis and effective algorithms for the capacitated vehicle routing problem.* PhD thesis, Northwestern University, Evanston, Illionios, 1999.

[8] N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operation Research Letters*, 34(1):58–68, 2006. doi: 10.1016/j.orl.2004.11.011.

[9] W.M. Carlyle, J.O. Royset, and R.K. Wood. Lagrangian relaxation and enumeration for solving constrained shortest-path problems. *Networks*, 51(3):155–170, 2008. doi: 10.1002/net.20212.

[10] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 33(10): 2972–2990, 2006. doi: 10.1016/j.cor.2005.02.029.

[11] N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for the vehicle routing problem based on spanning tree and shortest path relaxation. *Mathematical Programming*, 10:255–280, 1981.

[12] M. Dell'Amico, F. Maffioli, and P. Värbrand. On prize-collecting tours and the asymmetric travelling salesman problem. *International Transactions in Operations Research*, 2(3):297–308, 1995.

[13] I. Dumitrescu and N. Boland. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks*, 42(3):135–153, 2003. doi: 10.1002/net.10090.

[14] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004. doi: 10.1002/net.v44:3.

[15] D. Feillet, P. Dejax, and M. Genderau. Traveling salesman problems with profits. *Transportion Science*, 39(2):188–205, 2005.

[16] M. Fischetti, J. J. Salazar-Gonzalez, and P. Toth. Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10(2):133–148, 1998. ISSN 1526-5528. doi: 10.1287/ijoc.10.2.133.

[17] H. Gehring and J. Homberger. A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. Technical report, University of Jyväskylä, 1999.

[18] M. Gendreau, G. Laporte, and F. Semet. A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32 (4):263–273, 1998.

[19] M. Grötschel and W.R. Pulleyblank. Clique tree inequalities and the symmetric travelling salesman problem. *Mathematics of Operations Research*, 11:537–569, 1986.

[20] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56(2):497–511, 2008. doi: 10.1287/opre.1070.0449.

[21] G. Laporte and S. Martello. The selective travelling salesman problem. *Discrete Apllied Mathematics*, 26:193–207, 1990.

[22] A.C. Leifer and M.B Rosenwein. Strong linear programming relaxations for the orienteering problem. *European Journal of Operational Research*, 73(3):517–523, 1994.

[23] A. N. Letchford, R. W. Eglese, and J. Lysgaard. Multistars, partial multistars and the capacitated vehicle routing problem. *Mathematical Programming, Series B*, 94(1):21–40, 2002.

[24] A.N. Letchford and J.J. Salazar-Gonzalez. Projection results for vehicle routing. *Math. Program.*, 105(2-3):251–274, 2006.

[25] R. Muhandiramge and N. Boland. Simultaneous solution of lagrangean dual problems interleaved with preprocessing for the weight constrained shortest path problem. *Networks*, 2009. doi: 10.1002/net20292.

[26] M. Padberg and M.R. Rao. Odd minimum cut-sets and $b$-matchings. *Mathematics of Operations research*, 7:67–80, 1982.

[27] M. Padberg and G. Rinaldi. Facet identification for the symmetric traveling salesman problem. *Mathematical Programming*, 47:219–257, 1990.

[28] B. Petersen, D. Pisinger, and S. Spoorendonk. Chvátal-Gomory rank-1 cuts used in a Dantzig-Wolfe decomposition of the vehicle routing problem with time windows. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 397–420. Springer, 2008. doi: 10.1007/978-0-387-77778-8_18.

[29] G. Righini and M. Salani. Symmetry helps: bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006. doi: 10.1016/j.disopt.2006.05.007.

[30] G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained shortest path problem. *Networks*, 51(3):155–170., 2008. doi: 10.1002/net.20212.

[31] P.D. Seymour. Sums of circuits. In W. T. Tutte J. A. Bondy, U.S.R. Murty, editor, *Graph Theory and Related Topics*, pages 341–355. Academic Press, New York, 1979.

[32] M. M. Solomon. Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research*, 35(2): 234–265, 1987. doi: 10.1287/opre.35.2.254.

[33] S. Spoorendonk and G. Desaulniers. Clique inequalities applied to vehicle routing problem with time windows. *INFOR*, 1:53–67, 2008.

[34] P. Toth and D. Vigo, editors. *The Vehicle Routing Problem.* SIAM, 2002.

[35] L. A. Wolsey. *Integer Programming.* John Wiley & Sons, Inc., 1998.

## A.1 Detailed Results

### A.1.1 Dynamic Programming Algorithm

This sections holds the detailed results for running the dynamic programming algorithms that we use for comparison with the BAC algorithm. The purpose of this section is to justify that our implementation is a state-of-the-art algorithm in this area. Our implementation is similar to the dynamic programming algorithms presented by Righini and Salani [30], but make use of the bounding function based on the relaxation of 2-cycle elimination dynamic programming algorithms and the fact that the instances are symmetric which makes it sufficient to only investigate half of the state space before starting to merge labels, see e.g., Baldacci et al. [2].

Tables A.8, A.10 and A.9 contains detailed information on the dynamic programming algorithms used for solving the instances in Sets 1, 2 and 3, respectively. The tables report the instance name *Instance*, and for the exact dynamic programming (*Exact DP*) and the decremental state-space (*DSSR DP*) dynamic programming algorithms we report the upper bound (*UB*), the lower bound (*LB*), the total number of states explored in thousands (*States*), the percentage of states fathomed by bounding using the lower bounds calculated by a 2-cycle elimination dynamic programming algorithm (*%Fat1*), by capacity infeasibility (*%Fat2*), and by dominance (*%Dom1*), and the total time to solve the problem (*Time*). For the decremental state-space dynamic programming we also report the number of iterations (*It*). If the instance could not be solved within the time limit of 3 hours a dash (*'-'*) entry is found in the *Time* column. Furthermore, in Table A.8 we supply the running time of the algorithms used by Righini and Salani [30] in columns (*Righini and Salani [30] Time*). These algorithms are run on the same machine as our implementation. We did not run the algorithms on the Set 2 and 3 instances since the results on the instances in Set 1 suggest that our implementation is superior. Furhtermore, for Set 3 the instances require a minimum of delivered demand on the path which is not supported by that implementation.

From A.8 we can see that our implementation of the dynamic programming algorithms is faster than that of Righini and Salani [30] and we are able to solve one more instances with the exact dynamic programming. Also we note that the percentage of states fathom by the bounding function (*%Fat1*) is quite small on the hard instances. This suggests that even without this feature our implementation may be faster. The comparison with the algorithms of Righini and Salani [30] suggests that our implementation may indeed be considered a state-of-the-art algorithm and that the comparison with the BAC algorithm is a fair one. From Table A.9 it is seen that we were not able to solve any of the Set 2 instances. Also, the number of labels fathomed is very low which suggest that the bounding function is not so

good. Improvements may be gained by trying stronger relaxations such as the *ng*-paths, see Baldacci et al. [3]. however, this is out of the scope of this paper. From Table A.10 we see that the percentage of states fathomed by bounding has increased dramatically compared to the Set 1 instances. This is suggests that the capacity of the vehicle is less binding on the Set 3 instances that on the Set 1 instances.

| | Exact DP | | | | | | | [30] | DSSR DP | | | | | | | | [30] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance | LB | UB | States | %Fat1 | %Fat2 | %Dom | Time | Time | LB | UB | States | %Fat1 | %Fat2 | %Dom | It | Time | Time |
| c_100_06 | -32 | -32 | 110 | 19.7 | 78.6 | 0.8 | 0.02 | 1.23 | -32 | -32 | 45 | 91.9 | 0.0 | 5.5 | 2 | 0.04 | 0.30 |
| c_100_07 | -41 | -41 | 594 | 15.3 | 82.7 | 1.0 | 0.06 | 1.62 | -41 | -41 | 86 | 84.9 | 0.0 | 13.0 | 2 | 0.05 | 0.45 |
| c_100_08 | -48 | -48 | 888 | 16.7 | 81.0 | 1.3 | 0.10 | 69.88 | -48 | -48 | 105 | 79.7 | 0.0 | 18.3 | 2 | 0.06 | 2.12 |
| c_100_09 | -57 | -57 | 5254 | 11.3 | 86.3 | 1.3 | 1.90 | 80.14 | -57 | -57 | 182 | 72.8 | 0.0 | 25.4 | 2 | 0.11 | 2.24 |
| c_100_10 | -64 | -64 | 7324 | 12.6 | 84.6 | 1.8 | 2.95 | 4189.63 | -64 | -64 | 616 | 70.7 | 0.0 | 27.2 | 4 | 0.55 | 4.05 |
| r_100_06 | -61 | -61 | 8074 | 6.1 | 88.6 | 4.5 | 2.74 | 154.56 | -61 | -61 | 1615 | 36.8 | 0.0 | 59.3 | 4 | 6.62 | 11.72 |
| r_100_07 | -67 | -67 | 26992 | 5.0 | 88.1 | 6.0 | 109.4 | 2149.72 | -67 | -67 | 1479 | 22.5 | 0.0 | 74.0 | 2 | 9.12 | 22.73 |
| r_100_08 | -77 | -77 | 82896 | 3.8 | 87.5 | 7.7 | 829.71 | - | -77 | -77 | 2156 | 12.6 | 0.0 | 84.5 | 2 | 14.77 | 41.08 |
| r_100_09 | -249 | -52 | 52806 | 9.3 | 48.8 | 37.5 | - | - | -86 | -86 | 3217 | 6.0 | 0.0 | 91.3 | 2 | 29.82 | 166.41 |
| r_100_10 | -277 | -55 | 31277 | 7.8 | 13.2 | 70.8 | - | - | -93 | -93 | 15475 | 2.8 | 0.0 | 94.4 | 4 | 395.64 | 1183.68 |
| rc_100_06 | -33 | -33 | 48 | 29.2 | 69.0 | 0.8 | 0.01 | 0.48 | -33 | -33 | 16 | 93.6 | 0.0 | 3.6 | 1 | 0.02 | 0.28 |
| rc_100_07 | -34 | -34 | 111 | 32.9 | 64.8 | 1.3 | 0.01 | 1.55 | -34 | -34 | 75 | 92.7 | 0.0 | 5.0 | 2 | 0.05 | 0.94 |
| rc_100_08 | -42 | -42 | 269 | 32.3 | 65.2 | 1.5 | 0.02 | 4.58 | -42 | -42 | 82 | 90.9 | 0.0 | 6.9 | 1 | 0.05 | 1.03 |
| rc_100_09 | -51 | -51 | 635 | 32.5 | 64.6 | 1.8 | 0.04 | 14.06 | -51 | -51 | 146 | 88.2 | 0.0 | 9.8 | 1 | 0.08 | 1.78 |
| rc_100_10 | -53 | -53 | 1299 | 37.9 | 59.0 | 2.1 | 0.10 | 59.91 | -53 | -53 | 238 | 86.2 | 0.0 | 12.0 | 1 | 0.13 | 11.29 |

Table A.8: Detailed results for the dynamic programming algorithm on Set 1.

| | Exact DP | | | | | | | DSSR DP | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance | LB | UB | States | %Fat1 | %Fat2 | %Dom | Time | LB | UB | States | %Fat1 | %Fat2 | %Dom | It | Time |
| c1_2_a | -15.299 | -2.794 | 19781 | 0.0 | 0.0 | 78.1 | - | -13.532 | -6.847 | 118770 | 0.0 | 0.0 | 98.8 | 4 | - |
| c1_4_a | -16.575 | -2.623 | 29274 | 0.0 | 0.0 | 81.5 | - | -14.190 | -5.098 | 320109 | 0.0 | 0.0 | 99.3 | 6 | - |
| c1_6_a | -15.215 | -2.403 | 60683 | 0.0 | 0.0 | 88.6 | - | -13.950 | - | 547416 | 0.1 | 0.0 | 99.6 | 6 | - |
| c1_8_a | -16.465 | -2.509 | 104296 | 0.0 | 0.0 | 93.9 | - | -14.291 | -5.520 | 866584 | 1.0 | 0.0 | 98.8 | 6 | - |
| c1_10_a | -16.785 | -2.262 | 156176 | 0.8 | 0.0 | 94.5 | - | -14.174 | -6.241 | 838509 | 9.7 | 0.0 | 90.1 | 3 | - |
| c2_2_a | -57.842 | -3.642 | 18369 | 0.0 | 0.0 | 74.9 | - | -52.275 | - | 162707 | 0.0 | 0.0 | 99.1 | 3 | - |
| c2_4_a | -54.540 | -2.724 | 28290 | 0.0 | 0.0 | 77.9 | - | -53.695 | - | 239914 | 0.0 | 0.0 | 99.6 | 2 | - |
| c2_6_a | -59.668 | -2.325 | 56006 | 0.0 | 0.0 | 87.1 | - | -54.635 | - | 535658 | 0.0 | 0.0 | 99.8 | 3 | - |
| c2_8_a | -58.401 | -2.776 | 87843 | 0.0 | 0.0 | 92.1 | - | -54.866 | - | 600954 | 0.0 | 0.0 | 99.9 | 2 | - |
| c2_10_a | -58.444 | -2.052 | 119843 | 0.0 | 0.0 | 94.1 | - | -54.010 | - | 851238 | 0.0 | 0.0 | 99.9 | 3 | - |
| r1_2_a | -48.322 | -3.230 | 25954 | 0.0 | 0.0 | 83.6 | - | -18.603 | -4.855 | 88279 | 0.0 | 0.0 | 97.7 | 2 | - |
| r1_4_a | -54.802 | -3.176 | 37262 | 0.0 | 0.0 | 82.8 | - | -34.649 | - | 278862 | 0.0 | 0.0 | 99.7 | 2 | - |
| r1_6_a | -56.717 | -3.326 | 55379 | 0.0 | 0.0 | 86.1 | - | -54.791 | - | 399680 | 0.0 | 0.0 | 99.7 | 2 | - |
| r1_8_a | -37.915 | -1.759 | 125817 | 0.0 | 0.0 | 93.6 | - | -31.407 | - | 710623 | 0.0 | 0.0 | 99.8 | 3 | - |
| r1_10_a | -26.964 | -1.370 | 169050 | 0.0 | 0.0 | 95.2 | - | -24.767 | - | 981713 | 0.0 | 0.0 | 99.8 | 3 | - |
| r2_2_a | -282.347 | -3.269 | 19402 | 0.0 | 0.0 | 76.4 | - | -189.270 | - | 144439 | 0.0 | 0.0 | 99.4 | 2 | - |
| r2_4_a | -207.490 | -2.643 | 40033 | 0.0 | 0.0 | 83.3 | - | -207.490 | - | 214033 | 0.0 | 0.0 | 99.7 | 1 | - |
| r2_6_a | -404.463 | -3.883 | 45094 | 0.0 | 0.0 | 84.3 | - | -404.463 | - | 308388 | 0.0 | 0.0 | 99.7 | 1 | - |
| r2_8_a | -173.592 | -2.182 | 116400 | 0.0 | 0.0 | 93.0 | - | -173.592 | - | 426781 | 0.0 | 0.0 | 99.8 | 1 | - |
| r2_10_a | -231.424 | -1.555 | 167671 | 0.0 | 0.0 | 94.9 | - | -231.424 | - | 521914 | 0.0 | 0.0 | 99.9 | 1 | - |
| rc1_2_a | -42.921 | -3.168 | 20358 | 0.0 | 0.0 | 76.1 | - | -16.320 | - | 119510 | 0.0 | 0.0 | 99.1 | 2 | - |
| rc1_4_a | -21.040 | -2.508 | 40743 | 0.0 | 0.0 | 84.3 | - | -16.699 | -4.780 | 326074 | 0.0 | 0.0 | 99.5 | 3 | - |
| rc1_6_a | -19.740 | -2.713 | 72255 | 0.0 | 0.0 | 89.3 | - | -17.342 | - | 523224 | 0.0 | 0.0 | 99.7 | 3 | - |
| rc1_8_a | -50.544 | -1.950 | 101377 | 0.0 | 0.0 | 92.3 | - | -50.544 | - | 441726 | 0.0 | 0.0 | 99.8 | 1 | - |
| rc1_10_a | -23.243 | -2.071 | 164462 | 0.0 | 0.0 | 95.0 | - | -16.481 | - | 1189521 | 0.5 | 0.0 | 99.3 | 4 | - |
| rc2_2_a | -217.864 | -3.168 | 20358 | 0.0 | 0.0 | 76.1 | - | -217.864 | - | 105586 | 0.0 | 0.0 | 99.4 | 1 | - |
| rc2_4_a | -107.968 | -2.508 | 40670 | 0.0 | 0.0 | 84.3 | - | -107.968 | - | 214393 | 0.0 | 0.0 | 99.7 | 1 | - |
| rc2_6_a | -104.490 | -2.713 | 72232 | 0.0 | 0.0 | 89.3 | - | -104.490 | - | 300165 | 0.0 | 0.0 | 99.8 | 1 | - |
| rc2_8_a | -260.072 | -1.950 | 101427 | 0.0 | 0.0 | 92.3 | - | -260.072 | - | 424859 | 0.0 | 0.0 | 99.8 | 1 | - |
| rc2_10_a | -135.825 | -2.071 | 164262 | 0.0 | 0.0 | 95.0 | - | -135.825 | - | 542635 | 0.0 | 0.0 | 99.9 | 1 | - |

Table A.9: Detailed results for the dynamic programming algorithm on Set 2.

| | Exact DP | | | | | | | DSSR DP | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance | LB | UB | States | %Fat1 | %Fat2 | %Dom | Time | LB | UB | States | %Fat1 | %Fat2 | %Dom | It | Time |
| E-n76-k7_a | -6.032 | -6.032 | 703 | 60.1 | 38.3 | 0.2 | 0.10 | -6.032 | -6.032 | 300 | 96.2 | 0.0 | 1.9 | 2 | 0.12 |
| E-n76-k7_b | -0.846 | -0.846 | 37 | 69.4 | 29.0 | 0.2 | 0.04 | -0.846 | -0.846 | 126 | 97.7 | 0.0 | 0.4 | 5 | 0.09 |
| E-n76-k8_a | -6.635 | -6.635 | 368 | 54.9 | 43.6 | 0.1 | 0.24 | -6.635 | -6.635 | 370 | 97.2 | 0.0 | 0.5 | 2 | 0.37 |
| E-n76-k8_b | -0.001 | -0.001 | 13 | 71.0 | 27.6 | 0.0 | 0.28 | -0.001 | -0.001 | 101 | 98.1 | 0.0 | 0.0 | 9 | 0.32 |
| E-n76-k10_a | -3.810 | -3.810 | 106 | 53.2 | 45.4 | 0.0 | 0.15 | -3.810 | -3.810 | 54 | 97.3 | 0.0 | 0.3 | 1 | 0.32 |
| E-n76-k10_b | 0.000 | 0.000 | 17 | 55.6 | 43.0 | 0.0 | 0.28 | 0.000 | 0.000 | 131 | 97.5 | 0.0 | 0.1 | 13 | 0.33 |
| E-n76-k14_a | -3.788 | -3.788 | 50 | 42.5 | 56.1 | 0.0 | 0.12 | -3.788 | -3.788 | 22 | 96.8 | 0.0 | 0.1 | 1 | 0.13 |
| E-n76-k14_b | -0.002 | -0.002 | 10 | 45.8 | 52.9 | 0.0 | 0.11 | -0.002 | -0.002 | 82 | 97.2 | 0.0 | 0.0 | 15 | 0.15 |
| E-n101-k8_a | -35.988 | - | 64845 | 95.3 | 0.0 | 2.2 | - | -23.977 | -23.977 | 4400 | 83.8 | 0.0 | 15.1 | 2 | 5.06 |
| E-n101-k8_b | -0.006 | -0.006 | 410 | 83.9 | 14.9 | 0.2 | 0.15 | -0.006 | -0.006 | 4178 | 98.4 | 0.0 | 0.4 | 17 | 0.79 |
| E-n101-k14_a | -6.667 | -6.667 | 396 | 51.1 | 47.7 | 0.1 | 0.05 | -6.667 | -6.667 | 369 | 97.2 | 0.0 | 1.2 | 3 | 0.1 |
| E-n101-k14_b | -0.002 | -0.002 | 62 | 57.9 | 41.0 | 0.0 | 0.03 | -0.002 | -0.002 | 601 | 98.1 | 0.0 | 0.2 | 17 | 0.16 |
| F-n45-k4_a | -1.001 | -1.001 | 17 | 83.9 | 13.1 | 0.3 | 1.29 | -1.001 | -1.001 | 55 | 97.0 | 0.0 | 0.4 | 3 | 1.54 |
| F-n72-k4_a | 0.006 | 0.006 | 22 | 93.6 | 4.7 | 0.1 | 102.06 | 0.006 | 0.006 | 327 | 97.9 | 0.0 | 0.3 | 6 | 102.83 |
| F-n135-k7_a | -76.778 | - | 12931 | 40.0 | 0.0 | 32.0 | - | -76.778 | 0.000 | 64075 | 38.0 | 0.0 | 60.9 | 1 | - |
| M-n121-k7_a | -109.056 | - | 16831 | 76.1 | 0.0 | 8.6 | - | -30.903 | - | 85715 | 73.6 | 0.0 | 25.0 | 3 | - |
| M-n121-k7_b | -14.967 | - | 101728 | 97.3 | 0.0 | 0.9 | - | -2.929 | - | 408527 | 96.0 | 0.0 | 2.9 | 14 | - |
| M-n151-k12_a | -5.799 | -5.799 | 12194 | 68.9 | 30.3 | 0.2 | 3.40 | -5.799 | -5.799 | 2485 | 97.5 | 0.0 | 1.6 | 2 | 0.77 |
| M-n151-k12_b | -0.002 | -0.002 | 1036 | 76.6 | 22.5 | 0.2 | 0.33 | -0.002 | -0.002 | 12331 | 98.7 | 0.0 | 0.5 | 25 | 2.18 |
| M-n200-k16_a | -4.648 | -4.648 | 39690 | 57.3 | 42.1 | 0.1 | 98.23 | -4.648 | -4.648 | 19336 | 98.6 | 0.0 | 0.7 | 3 | 36.05 |
| M-n200-k16_b | -0.001 | -0.001 | 17595 | 76.0 | 23.4 | 0.1 | 20.84 | -0.001 | -0.001 | 105539 | 98.7 | 0.0 | 0.7 | 30 | 118.23 |
| M-n200-k17_a | -5.762 | -5.762 | 199 | 69.7 | 29.7 | 0.0 | 0.08 | -5.762 | -5.762 | 470 | 99.3 | 0.0 | 0.1 | 4 | 0.12 |
| M-n200-k17_b | -0.001 | -0.001 | 7909 | 73.5 | 25.9 | 0.1 | 2.25 | -0.001 | -0.001 | 49836 | 98.7 | 0.0 | 0.6 | 34 | 7.08 |
| P-n70-k10_a | -2.852 | -2.852 | 67 | 47.8 | 50.7 | 0.0 | 0.11 | -2.852 | -2.852 | 33 | 96.8 | 0.0 | 0.2 | 1 | 0.11 |
| P-n70-k10_b | -0.001 | -0.001 | 11 | 49.9 | 48.6 | 0.0 | 0.10 | -0.001 | -0.001 | 98 | 97.1 | 0.0 | 0.0 | 14 | 0.22 |
| P-n76-k4_a | -2.903 | -2.903 | 1177 | 66.5 | 31.9 | 0.0 | 1.69 | -2.903 | -2.903 | 1381 | 97.6 | 0.0 | 0.5 | 2 | 4.35 |
| P-n76-k4_b | 0.000 | 0.000 | 388 | 78.6 | 19.9 | 0.0 | 1.38 | 0.000 | 0.000 | 3137 | 98.1 | 0.0 | 0.1 | 10 | 19.63 |
| P-n76-k5_a | -3.959 | -3.959 | 885 | 61.8 | 36.7 | 0.1 | 0.58 | -3.959 | -3.959 | 3330 | 97.4 | 0.0 | 0.6 | 7 | 2.48 |
| P-n76-k5_b | -0.001 | -0.001 | 173 | 76.0 | 22.5 | 0.0 | 0.95 | -0.001 | -0.001 | 1890 | 98.1 | 0.0 | 0.1 | 14 | 5.39 |
| P-n101-k4_a | -15.353 | - | 128676 | 97.3 | 0.0 | 0.4 | - | -7.219 | -7.219 | 180893 | 94.9 | 0.0 | 3.8 | 6 | 3455.28 |
| P-n101-k4_b | -19.816 | - | 124105 | 97.9 | 0.0 | 0.2 | - | -0.643 | - | 416480 | 97.2 | 0.0 | 1.7 | 11 | - |

Table A.10: Detailed results for the dynamic programming algorithm on Set 3.

# Chapter 5 Partial Path Column Generation for the Vehicle Routing Problem

Mads Kehlet Jepsen, Bjørn Petersen, David Pisinger

*DTU Management Engineering, Technical University of Denmark Produktionstorvet 424, DK-2800 Kongens Lyngby, Denmark*

Marts 2010

## Abstract

This paper presents a column generation algorithm for the Capacitated Vehicle Routing Problem (CVRP) and the Vehicle Routing Problem with Time Windows (VRPTW). Traditionally, column generation models of the CVRP and VRPTW have consisted of a Set Partitioning master problem with each column representing a route. The use of Elementary routes, where no customer is visited more than once, have shown superior results for both CVRP and VRPTW. However, algorithms for solving the pricing problems do not scale well when the number of feasible routes increases. We suggest to relax the constraint that 'each column is a route' into 'each column is a part of the giant tour'; a so-called partial path, i.e., not necessarily starting and ending in the depot. This way, the length of the partial path can be bounded and a better control of the size of the solution space for the pricing problem can be obtained. It is shown that the LP-relaxed partial path formulation gives a tighter bound than the LP-relaxation of a 2-index formulation, and in some cases it is even tighter than the bound found by classical decomposition into routes.

## 5.1 Introduction

The *Capacitated Vehicle Routing Problem* (CVRP) can be described as follows: A set of customers $C$ having a demand $d_i$, needs to be serviced by a number of vehicles all starting and ending at a central depot. Each customer must be visited exactly once and the capacity $Q$ of the vehicles may not be exceeded. The objective is to service all customers traveling the least possible distance. In this paper we consider a homogeneous fleet, i.e., all vehicles are identical. The *Vehicle Routing Problem with Time Windows* (VRPTW) extends the CVRP by imposing that each customer must be visited within a given time window. We will use the term VRP to denote Vehicle Routing Problems with time and/or capacity constraints.

The standard Dantzig-Wolfe decomposition of the arc flow formulation of the VRP is to split the problem into a master problem formulated as a Set Partitioning Problem, and a pricing problem formulated as an Elementary Shortest Path Problem with Resource Constraints (ESPPRC), where capacity (and time) are the constrained resources. A restricted master problem can be solved with delayed column generation and embedded in a branch-and-bound framework to ensure integrality. Applying cutting planes either in the master or the pricing problem leads to a Branch-and-Cut-and-Price algorithm (BCP). Kohl et al. [25] implemented a successful BCP algorithm for the VRPTW by applying *sub-tour elimination* constraints and *two-path* cuts, Cook and Rich [10] generalized the *two-path* cuts to $k$*-path* cuts, and Fukasawa et al. [19] applied a range of valid inequalities for the CVRP based on the branch and cut algorithm of Lysgaard et al. [28]. Common for these BCP algorithms is that all applied cuts are valid inequalities for the VRPTW respectively the CVRP with regard to the *original* arc flow formulation, and have a structure which makes it possible to handle values of the dual variables in the pricing problem without increasing the complexity of the problem. Fukasawa et al. [19] refer to this as a *robust* approach in their paper. The topic of column generation and BCP algorithms has been surveyed by Barnhart et al. [4] and Lubbecke and Desrosiers [26]. Recently the BCP framework was extended to include valid inequalities for the master problem, more specifically by applying the subset row (SR) inequalities to the Set Partitioning master problem in Jepsen et al. [23] and later by applying Chvátal-Gomory Rank-1 (CG1) inequalities in Petersen et al. [29]. Desaulniers et al. [13] solved several unsolved instances by adding generalized k-Path inequlities and generated columns heuristically using a tabu search and finally introduced a new algorithm to solve the pricing problem where partial elementarity is used. Baldacci et al. [2] improved the lower bound by adding strengthened capacity inequalities and clique inequalities to an algorithm where columns with potentially negative reduced cost are enumerated (after good upper and lower bounds are found). Dror [16] showed that the ESPPRC, with time and capacity constraints, is strongly $\mathcal{NP}$-hard. Hence, a relaxation of the ESPPRC was used as the pricing problem in earlier BCP approaches for the VRPTW. The relaxed pricing problem where non-elementary paths are allowed is denoted the Shortest Path Problem with Resource Constraints (SPPRC) and can be solved in pseudo-polynomial time by dynamic programming using for instance a labeling algorithm, see Desrochers [14]. Considering a single capacity resource Christofides et al. [9] suggested to remove 2-cycles from the paths. This was later generalized to the variant with time windows by Desrochers et al. [15]. Irnich and Villeneuve [22] extended the framework further to $k$-cycle elimination ($k$-cyc-SPPRC), where cycles containing $k$ or less nodes are forbidden.

Beasley and Christofides [5] proposed to solve the ESPPRC using Lagrangian relaxation. However, labeling algorithms have recently become the most popular approach to solve the ESPPRC, see e.g. Dumitrescu [17] and Feillet et al. [18]. When solving the ESPPRC with a labeling algorithm, a binary resource for each

node is added, increasing the complexity of the algorithm compared to the solution of the SPPRC or the $k$-cyc-SPPRC. Righini and Salani [30] developed a labeling algorithm using the idea of Dijkstra's bi-directional shortest path algorithm that expands both forward and backward from the depot and connects routes in the middle, thereby potentially reducing the running time of the algorithm. Furthermore, Righini and Salani [31] and Boland et al. [6] proposed a decremental state space algorithm that iteratively solves a SPPRC, by iteratively applying binary resources to force nodes to be visited at most once. Recently Chabrier [7], Danna and Pape [11], and Salani [32] successfully solved several previously unsolved instances of the VRPTW from the benchmarks of Solomon [33] using a labeling algorithm for the ESPPRC. However, these algorithms have some weaknesses when dealing with very long (measured in the number of visited nodes) paths, when resource constraints are not tight. Christofides and Eilon [8] introduced the giant-tour representation in which all the routes are represented by one single *giant* tour, i.e., all the routes are concatenated into a single tour.

In this paper we propose a decomposition approach based on the generation of partial paths and the concatenation of these. The main idea is to limit the solution space of the pricing problem by bounding a resource, e.g., the number of nodes on a path or the capacity on it. The master problem combines a known number of these bounded partial paths such that all customers are visited. In this way we get a better controle of the pricing problem. If the original pricing problem is too difficult to solve for each vehicle, we may imposing a limit on the nodes in a partial path. If the original pricing problem for each vehicle is easy, we can choose looser bounds such that the partial paths get longer and lead to tighter bounds.

The paper is organized as follows: In Section 5.2 we describe how to use the giant tour formulation of VRP to obtain the partial path formulation. Section 5.3 introduces a mathematical model based on partial paths. Section 5.4 shows how the model is decomposed through Dantzig-Wolfe decomposition, and describes how to calculate the reduced cost of columns in a delayed column generation framework. Section 5.5 describes how to use the load resource to divide the solution space. Section 5.8 concludes the paper discussing future work.

## 5.2 Bounded Partial Paths

Given a graph $G(V, A)$ with nodes $V = C \cup \{0\}$ and arcs $A$, where $C$ is the set of customers, and $0$ is the depot. Moreover, we have a set $R$ of resources which e.g. can be load and/or time. Each resource $r \in R$ has a resource window $[a_i^r, b_i^r]$ that must be met upon arrival to node $i \in V$, and a consumption $\tau_{ij}^r \geq 0$ for using arc $(i, j) \in A$. A resource consumption at a node $i \in C$ is modeled by a resource consumption at edge $(i, j)$, and hence usually $\tau_{0j}^r = 0$ for all $j \in C$. A global capacity limit $Q$ can be modeled by imposing a resource window $[0, Q]$ for the depot node 0.

The VRP can now be stated as: Find a set of routes starting and ending at the

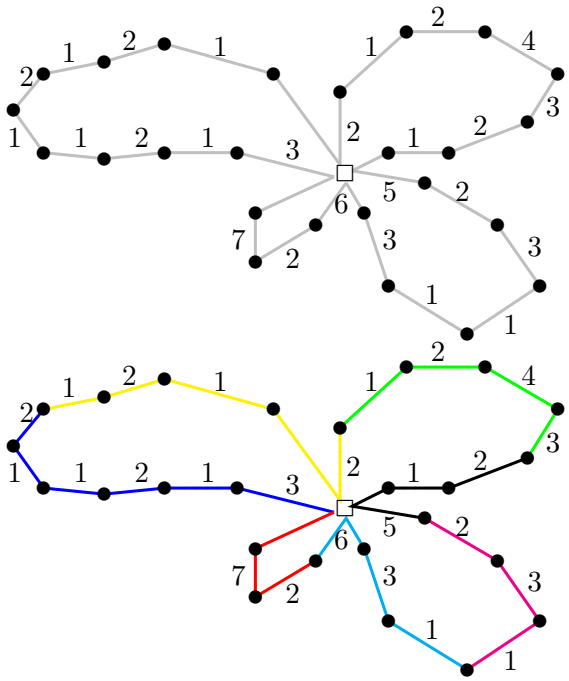


Figure 5.1: Giant-tour (upper) and corresponding giant-tour split into partial paths (lower), each bounded by the capacity $Q = 10$.

depot node $0$ satisfying all resource windows, such that the cost is minimized and all customers $C$ are visited.

A solution to the VRP will consist of a number of routes

$$\begin{aligned} &0 \to i_1^1 \to \ldots \to i_{k_1}^1 \to 0, \\ &0 \to i_1^2 \to \ldots \to i_{k_2}^2 \to 0, \\ &\vdots \\ &0 \to i_1^n \to \ldots \to i_{k_n}^n \to 0 \end{aligned}$$

where $n$ is the number of vehicles, and $k_j$ is the length of the $j$'th route. A natural decomposition of the VRP is to split the problem into these separate routes, where a master problem ensures that all customers are visited once. We will call this the *classical* decomposition. However, using the classical decomposition, the number of nodes in each individual route may vary a lot, making it difficult to solve some of the subproblems.

Instead we consider the giant-tour representation by Christofides and Eilon [8]

$$0 \to i_1^1 \to \ldots \to i_{k_1}^1 \to 0 \to i_1^2 \to \ldots \to i_{k_2}^2 \to 0 \to \ldots \to 0 \to i_1^n \to \ldots \to i_{k_n}^n \to 0$$

A giant-tour (see Figure 5.1) is one long path visiting all customers once and the depot several times. The consumption of resources $r \in R$ is reset each time the depot node is encountered. If we decompose the VRP into smaller segments of the giant-tour, we may to a larger extent controle that the number of nodes visited in

each partial path is of similar length. In this way we can balance the hardness of the subproblems (see Figure 5.1 for an illustration).

The decompostion is done by imposing an upper limit on a resource $r' \in R$, e.g., bounding the path length in the number of nodes for each partial path, or bounding the load. The giant tour introduced in Figure 5.1 can be decomposed into a number of partial paths by bounding a resource. In the following the number of visited customers in $C$ is considered to be the bounding resource. Bounding the load resource is a bit more complicated and will be addressed in Section 5.5.

Each segment represents a partial path of the giant-tour. With a fixed number of customers $L$ on each partial path, $K$ partial paths are needed to ensure that all customers are visited i.e., $L \cdot K \geq |C|$. The partial paths can start and end in any node in $V$ and it can visit the depot several times. A partial path could for example be:

$$i_1 \rightarrow i_2 \rightarrow 0 \rightarrow i_3 \rightarrow 0 \rightarrow i_4$$

In the following we will make a graph representation for the problem of finding the $K$ partial path of length $L$. This is done by replicating the graph $K$ times and connecting the replications by special arcs. Each of the replications is connected with arcs directed from one replication to a following replication. This leads to a layered graph with $K$ layers $1, .., K$ where there are no outgoing arcs of the final layer. Each layer $k \neq K$ is connected to the subsequent layer $k + 1$. Each pair of subsequent layers are connected with the set of arcs leaving node $i$ in layer $k \neq K$ and entering layer $k + 1$.

Consider the graph $G'(V', A')$ consisting of a set of layers $\mathcal{K} = \{1, \ldots, K\}$, each layer representing $G$ for a partial path. Let $G^k$ be the sub graph of $G'$ representing layer $k$ with node set $V^k = \{(i, k) : i \in V\}$ for all $k \in \mathcal{K}$ and arc set $A^k = \{(i, j, k) : (i, j) \in A\}$ for all $k \in \mathcal{K}$. Let $A^* = \{(i, i, k) : (i, k) \in V^k \wedge (i, k + 1) \in V^{k+1} \wedge k \in \mathcal{K}\}$ be the set of interconnecting arcs, i.e., the arcs connecting a layer $k$ with the layer above $k$ namely layer $k+1$ for all $k \in \mathcal{K}$ and all nodes $i \in V$ (layer $K + 1$ is defined to be layer $1 \in \mathcal{K}$ and layer 0 is defined to be layer $K \in \mathcal{K}$). Let $V' = \bigcup_{k \in \mathcal{K}} V^k$ and let $A' = \bigcup_{k \in \mathcal{K}} A^k \cup A^*$. An illustration of $G'$ can be seen in Figure 5.2. Note, that arcs $(i, i, k)$ are not present in $A^k$ and that arcs $(i, j, k)$ with $i \neq j$ are present in $A^*$, so all arcs $(i, j, k) \in A'$ can be uniquely indexed.

The resource consumption $\tau^r_{ij}$ of arcs $(i, j) \in A^k$ is the same as in the original graph $A$, hence we omit the index $k$. The resource consumption of interconnecting arcs $(i, j) \in A^*$ is $\tau^r_{ij} = 0$.

Let $L$ be the upper bound on the length of each partial path, and let $|C|$ be the length of the combined path (the giant-tour). Now, exactly $K = \lceil |C|/L \rceil$ partial paths are needed to form the combined path, since $L \lceil |C|/L \rceil \geq |C| > L (\lceil |C|/L \rceil - 1)$. Once $K$ has been calculated, we can further reduce the path length to $L = \lceil |C|/K \rceil$.

With the length of a path defined as the number of customers on it, the problem is now to find partial paths of length at most $L$ in $K$ layers with $L \cdot K \geq |C| >$

$L \cdot (K-1)$, so that each partial path $p$ ending in node $i \in V$ is met by another partial path $p'$ starting in $i$. All partial paths are combined while not visiting any customers more than once and satisfying all resource windows. A customer $i \in C$ is considered to be on a partial path $p$ if $i$ is visited on $p$ and is not the end node of $p$.

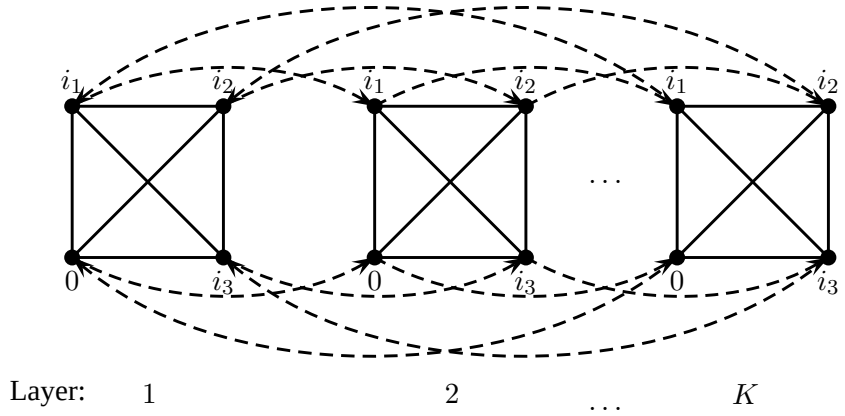



Figure 5.2: Illustration of $G'$ with $|C| = 3$, $K = 3$, and $L = 1$. Full-drawn lines represent two arcs; one in each direction. Dashed lines are the interconnecting arcs $A^*$.

## 5.3 The Vehicle Routing Problem

We present two models for the VRP problem defined in previous section. The 2-index model is most compact, while the 3-index model is better suited for decomposition.

**2-index formulation of the VRP** In the following let $c_{ij}$ be the cost of arc $(i,j) \in A$, $x_{ij}$ be the binary variable indicating the use of arc $(i,j) \in A$, and $T_{ij}^r$ (the resource stamp) be the consumption of resource $r \in R$ at the beginning of arc $(i,j) \in A$. Let $\delta^+(i)$ and $\delta^-(i)$ be the set of outgoing respectively ingoing arcs of node $i \in V$. Combining the two index model from Bard et al. [3] with the constraints ensuring the time windows for the ATSP by Ascheuer et al. [1] a

mathematical model can be formulated as follows:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{5.1}$$

$$\text{s.t.} \sum_{(i,j)\in\delta^+(i)} x_{ij} = 1 \qquad \forall i \in C \tag{5.2}$$

$$\sum_{(j,i)\in\delta^-(i)} x_{ji} = \sum_{(i,j)\in\delta^+(i)} x_{ij} \qquad \forall i \in V \tag{5.3}$$

$$\sum_{(j,i)\in\delta^-(i)} (T^r_{ji} + \tau^r_{ji}x_{ji}) \leq \sum_{(i,j)\in\delta^+(i)} T^r_{ij} \qquad \forall r \in R,\ \forall i \in C \tag{5.4}$$

$$a^r_i x_{ij} \leq T^r_{ij} \leq b^r_i x_{ij} \qquad \forall r \in R,\ \forall (i,j) \in A \tag{5.5}$$

$$T^r_{ij} \geq 0 \qquad \forall r \in R,\ \forall (i,j) \in A \tag{5.6}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{5.7}$$

The objective (5.1) sums up the cost of the used arcs. Constraints (5.2) ensure that each customer is visited exactly once, and (5.3) are the flow conservation constraints. Constraints (5.4) and (5.5) ensure the resource windows are satisfied. It is assumed that the bounds on the depot are always satisfied. Note, that no sub-tours can be present since only one resource stamp per arc exists and the arc weights are positive for all $(i,j) \in A : i \in C$.

For a one dimensional resource such as load the capacity constraints a stronger lower bound of the LP relaxation can be obtained by replacing (5.4) to (5.6) with $x(\delta^+(S)) \geq r(S)$, where $r(S)$ is a the minimum number of vehicles needed to service the set $S$. All though this model can not be directly solved it is possible to overcome this problem by only including the constraints that are violated. For more details on how to separate the constraint and calculate the value of $r(S)$ the reader is refered to Toth and Vigo [34].

**3-index formulation of the VRP** Let $x_{ij}^k$ be the variable indicating the use of arc $(i,j,k) \in A'$. Problem (5.1)–(5.7) is rewritten to:

$$\min \sum_{k\in\mathcal{K}} \sum_{(i,j)\in A} c_{ij} x_{ij}^k \tag{5.8}$$

$$\text{s.t.} \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\delta^+(i)} x_{ij}^k = 1 \qquad \forall i \in C \tag{5.9}$$

$$\sum_{(i,j)\in\delta^+(i)} x_{ij}^k \leq 1 \qquad \forall k \in \mathcal{K},\ \forall i \in C \tag{5.10}$$

$$\sum_{k\in\mathcal{K}} \left( x_{ii}^{k-1} + \sum_{(j,i)\in\delta^-(i)} x_{ji}^k \right) = \sum_{k\in\mathcal{K}} \left( x_{ii}^k + \sum_{(i,j)\in\delta^+(i)} x_{ij}^k \right) \qquad \forall i \in V \tag{5.11}$$

$$x_{ii}^{k-1} + \sum_{(j,i)\in\delta^-(i)} x_{ji}^k = x_{ii}^k + \sum_{(i,j)\in\delta^+(i)} x_{ij}^k \qquad \forall k \in \mathcal{K},\ \forall i \in V \tag{5.12}$$

$$\sum_{k\in\mathcal{K}} \sum_{i\in V} x_{ii}^k = K \tag{5.13}$$

$$\sum_{i\in C} \sum_{(i,j)\in A} x_{ij}^k \leq L \qquad \forall k \in \mathcal{K} \tag{5.14}$$

$$\sum_{k\in\mathcal{K}} \sum_{(j,i)\in\delta^-(i)} \left( T_{ji}^{rk} + \tau_{ji}^r x_{ji}^k \right) \leq \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\delta^+(i)} T_{ij}^{rk} \qquad \forall r \in R,\ \forall i \in C \tag{5.15}$$

$$\sum_{(j,i)\in\delta^-(i)} \left( T_{ji}^{rk} + \tau_{ji}^r x_{ji}^k \right) \leq \sum_{(i,j)\in\delta^+(i)} T_{ij}^{rk} \qquad \forall r \in R,\ \forall k \in \mathcal{K},\ \forall i \in C \tag{5.16}$$

$$a_i^r \sum_{k\in\mathcal{K}} x_{ij}^k \leq \sum_{k\in\mathcal{K}} T_{ij}^{rk} \leq b_i^r \sum_{k\in\mathcal{K}} x_{ij}^k \qquad \forall r \in R,\ \forall (i,j) \in A \tag{5.17}$$

$$a_i^r x_{ij}^k \leq T_{ij}^{rk} \leq b_i^r x_{ij}^k \qquad \forall r \in R,\ \forall k \in \mathcal{K},\ \forall (i,j) \in A \tag{5.18}$$

$$T_{ij}^{rk} \geq 0 \qquad \forall r \in R,\ \forall k \in \mathcal{K},\ \forall (i,j) \in A \tag{5.19}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall k \in \mathcal{K},\ \forall (i,j) \in A \tag{5.20}$$

The objective (5.8) sums up the cost of the used arcs. Constraints (5.9) ensure that all customers are visited exactly once, while the redundant constraints (5.10) ensure that no customer is visited more than once. Constraints (5.11) maintain flow conservation between the original nodes $V$, and can be rewritten as

$$\sum_{k\in\mathcal{K}}\sum_{(j,i)\in\delta^-(i)} x_{ji}^k = \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\delta^+(i)} x_{ij}^k \qquad \forall i \in V$$

since $\sum_{k\in\mathcal{K}} x_{ii}^{k-1} = \sum_{k\in\mathcal{K}} x_{ii}^k$. Constraints (5.12) maintain flow conservation within a layer. Constraint (5.13) ensures that $K$ partial paths are selected and constraints (5.14) that the length of the partial path in each layer is at most $L$. Constraints (5.15) connect the resource variables on a global level and constraints (5.16) connect the resource variables within each single layer. Note, that since constraints (5.15) and (5.16) are omitted for the depot, it is not constrained by resources. Constraints (5.17) globally enforce the resource windows and the redundant constraints (5.18) enforce the resource windows within each layer.

## 5.4 Dantzig-Wolfe Decomposition

We use Dantzig-Wolfe decompostion of the 3-index formulation of the VRP, defined in (5.8)–(5.20) to reach the following master and a pricing problem. In the process of the decomposition the $K$ identical pricing problems are combined into a single pricing problem.

### 5.4.1 Master Problem

Let $\lambda_p$ a binary variable indicating whether partial path $p$ is used. We use Dantzig-Wolfe decomposition where the constraints (5.9), (5.11), (5.13), (5.15), and (5.17) are kept in the master problem. Since the vehicles are identical, we can aggregate

over the sets $A^k$ getting the following master problem (PP):

$$\min \sum_{p \in P} c_p \lambda_p \tag{5.21}$$

$$\text{s.t.} \sum_{p \in P} \sum_{(i,j) \in \delta^+(i)} \alpha_{ij}^p \lambda_p = 1 \qquad \forall i \in C \tag{5.22}$$

$$\sum_{p \in P: e^p = i} \lambda_p = \sum_{p \in P: s^p = i} \lambda_p \qquad \forall i \in V \tag{5.23}$$

$$\sum_{p \in P} \lambda_p = K \tag{5.24}$$

$$\sum_{(j,i) \in \delta^-(i)} \left( T_{ji}^r + \sum_{p \in P} \tau_{ji}^r \alpha_{ji}^p \lambda_p \right) \leq \sum_{(i,j) \in \delta^+(i)} T_{ij}^r \qquad \forall r \in R,\ \forall i \in C \tag{5.25}$$

$$a_i^r \sum_{p \in P} \alpha_{ij}^p \lambda_p \leq T_{ij}^r \leq b_i^r \sum_{p \in P} \alpha_{ij}^p \lambda_p \qquad \forall r \in R,\ \forall (i,j) \in A \tag{5.26}$$

$$T_{ij}^r \geq 0 \qquad \forall r \in R,\ \forall (i,j) \in A \tag{5.27}$$

$$\lambda_p \in \{0,1\} \qquad \forall p \in P \tag{5.28}$$

In this formulation, $\alpha_{ij}^p$ is the number of times arc $(i,j) \in A$ is used on path $p \in P$ and $s^p$ and $e^p$ indicate the start respectively the end node of partial path $p \in P$. Constraints (5.22) ensure that each customer is visited exactly once. Constraints (5.23) link the partial paths together by flow conservation. Constraint (5.24) is the convexity constraint ensuring that $K$ partial paths are selected. Constraints (5.25) and (5.26) enforce the resource windows.

**Tightness of bounds:** Before we turn our attention to the pricing problem we prove the following theorems about the quality of the bounds obtained by the decomposition.

**Theorem 1.** *Let $z_{LP}$ be an LP-solution to* (5.1)–(5.7) *and let $z_{PP}$ be an LP-solution to* (5.21)–(5.28) *then $z_{LP} \leq z_{PP}$ for all instances of VRP.*

*Proof.* $z_{LP} \leq z_{PP}$ since all solutions to (5.21)–(5.28) map to solutions to (5.1)–(5.7). □

**Theorem 2.** *Let $z_{PP}$ as before be an LP-solution to* (5.21)–(5.28)*, and $z_{EP}$ be the LP-solution to the classical decomposition of VRP into an elementary route for each vehicle. Then instances exist where $z_{PP} > z_{EP}$.*

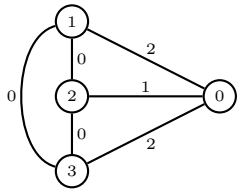



Figure 5.3: Three customers with demand of 1 and vehicle capacity $Q = 2$. Distances are indicated on the edges. There are six feasible routes $(\{0,1,0\}, \{0,2,0\}, \{0,3,0\}, \{0,1,2,0\}, \{0,1,3,0\}, \{0,2,3,0\})$ having the costs $(4,2,4,3,4,3)$. The LP solution is $(0,0,0,\frac{1}{2},\frac{1}{2},\frac{1}{2})$ with objective $z_{EP} = 5$. Using the partial path formulation with max path length $L = 3$ and $K = 1$ we find the optimal solution $(\{0,1,3,0,2,0\})$ with objective $z_{PP} = 6$.

*Proof.* An instance with $z_{PP} > z_{EP}$ can be constructed with three customers each with a demand of 1 and vehicle capacity $Q = 2$. Using a max path length of $L = 3$, we find $z_{PP} = 6$ while $z_{EP} = 5$. (See Figure 5.3). □

### 5.4.2 Pricing Problem

The $K$ pricing problems corresponding to the master problem (5.21)–(5.28) are defined by constraints (5.10), (5.12), (5.14), (5.16), and (5.18) and can be formulated as a single ESPPRC where the depot is allowed to be visited more than once. Let $s$ and $e$ be a super source respectively a super target node. Arcs $(s,i)$ and $(i,e)$ for

all $i \in V$ are added to $G$ with cost and resource consumption 0.

$$\min \sum_{(i,j)\in A} \overline{c}_{ij} x_{ij} \tag{5.29}$$

$$\text{s.t.} \sum_{(s,i)\in\delta^+(s)} x_{si} = 1 \tag{5.30}$$

$$\sum_{(i,e)\in\delta^-(e)} x_{ie} = 1 \tag{5.31}$$

$$\sum_{(i,j)\in A} x_{ij} \leq 1 \qquad \forall i \in C \tag{5.32}$$

$$\sum_{(j,i)\in\delta^-(i)} x_{ji} = \sum_{(i,j)\in\delta^+(i)} x_{ij} \qquad \forall i \in V \tag{5.33}$$

$$\sum_{(i,j)\in A} \tau^{r'}_{ij} x_{ij} \leq L \tag{5.34}$$

$$\sum_{(j,i)\in\delta^-(i)} (T^r_{ji} + \tau^r_{ji} x_{ji}) \leq \sum_{(i,j)\in\delta^+(i)} T^r_{ij} \qquad \forall r \in R,\ \forall i \in C \tag{5.35}$$

$$a^r_i x_{ij} \leq T^r_{ij} \leq b^r_i x_{ij} \qquad \forall r \in R,\ \forall (i,j) \in A \tag{5.36}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{5.37}$$

The objective (5.29) minimizes the reduced cost of a column in (PP). Constraints (5.30) and (5.31) ensure that the path starts in $s$ respectively ends in $e$. Constraints (5.32) dictates that no node is visited more than once, thereby ensuring elementarity. Constraints (5.33) conserve the flow. Constraint (5.34) ensures that the partial path does not use more than the allowed amount $L$ of the restricted resource $r'$. Constraints (5.35) and (5.36) ensure the resource windows are satisfied for all customers. Note, since constraints (5.35) hold for $i \in U$ (excluding the depot), a resource is only restricted by its lower limit $a^r_0$ for all $r \in R$ each time a path leaves the depot.

Let $\pi$ ($\pi_i \geq 0 : \forall i \in C$) be the duals of (5.22) and $\pi_0 = 0$, let $\mu$ be the duals of (5.23), let $\beta \leq 0$ be the dual of (5.24), let $\nu$ ($\nu \leq 0 : \forall i \in C$) be the duals of (5.25) and $\nu_0 = 0$, and let $\underline{\omega} \leq 0$ and $\overline{\omega} \geq 0$ be the dual of (5.26). Let $A_C = A \setminus (\delta^+(s) \cup \delta^-(e))$, the cost of the arcs in this ESPPRC are then given as:

$$\overline{c}_{ij} = c_{ij} - \beta + \begin{cases} c_{ij} - \pi_i - \tau_{ij}\nu_j - \sum_{r\in R} a^r_i \underline{\omega}^r_i + \sum_{r\in R} b^r_i \overline{\omega}^r_i & \forall (i,j) \in A_C \\ \mu_j & \forall (s,j) \in \delta^+(s) \\ -\mu_i & \forall (i,e) \in \delta^-(e) \end{cases}$$

The pricing problem is now an to find an elementary shortest path from $s$ to $e$.

**Solving the pricing problem:** ESPPRCs can be solved by various labeling algorithms, see e.g. Desaulniers et al. [12], Irnich [20], Irnich and Desaulniers [21], and Righini and Salani [30].

**Branching:** Integrality can be obtained by branching on the original variables, which can be accomplished by cuts in the master problem (see Vanderbeck [35]), e.g., let $X_{ij}$ be the set of partial paths that utilize arc $(i, j)$ then the branch rule $x_{ij} = 0 \ \vee \ x_{ij} = 1$ can be expressed by the dichotomy:

$$\sum_{p \in X_{ij}} \lambda_p = 0 \ \vee \ \sum_{p \in X_{ij}} \lambda_p = 1.$$

## 5.5 Bounding the Load Resource

The giant tour introduced in Section 5.1 can be decomposed into a number of partial paths by bounding a resource $r'$, e.g. the number of nodes, the time, or the load. In this section we consider the latter. The load constraint is present in CVRP and VRPTW and is a special type of resource constraints. If $Q$ is the maximal load of a vehicle and $d_i : i \in C$ is the demand of the costumers, then the accumulated demand on a route may not exceed $Q$. The goal is that equation (5.34) is expressed on the form:

$$\sum_{(i,j) \in A} d_i x_{ij} \leq L$$

where $L$ is a given threshold value for the load resource. This will potentially lead to an easier pricing problem. For dynamic programming based algorithms the complexity is dependent on the size of $L$. In the length case we rounded up the expression $|C|/K$ to ensure feasibility. In the following we will discuss a similar approach for bounding on the load resource.

Let the total demand of the customers be $D = \sum_{i \in C} d_i$. A lower bound on the number of partial paths needed is: $K = \lceil D/L \rceil$. However, we cannot just split the giant tour into $K$ partial paths of capacity $L$ since there is no guaranty that the optimal giant tour can be split into partial paths of equal capacity.

Let the largest demand be defined as $d_{\max} = \max_{i \in C} d_i$, and assume that $L \geq d_{\max}$. Then, we need to allow up to $d_{\max} - 1$ extra capacity in each partial path, to compensate for possibly uneven splitting. This means that for a given $K$ we find $L_{ub} = \lceil D/K \rceil + (d_{\max} - 1)$ as the upper bound on the resource consumption.

An alternative approach to increasing $L$ to $L_{ub}$ is to allow an additional edge exceeding $L$ to be selected in the pricing problem. This may complicate the pricing problem, though.

The remainder of this section addresses alternative strategies to avoid complicating the pricing problem. One such alternative is to introduce the concept of connector arcs. A connector arc is a single arc between two nodes which combines two partial paths. For each layer $k \in \mathcal{K}$ and original arc $(i, j) \in A$ there is connector arc to the subsequent layer.

Figure 5.4 illustrates the idea of the connector arcs. The dashed lines from node 0 in layer 1 orientated towards layer 2 to node $i_1, i_2$ and $i_3$, illustrates the connectors out of node 0 in layer 1. Similar nodes $i_1$ in layer one will have connectors to

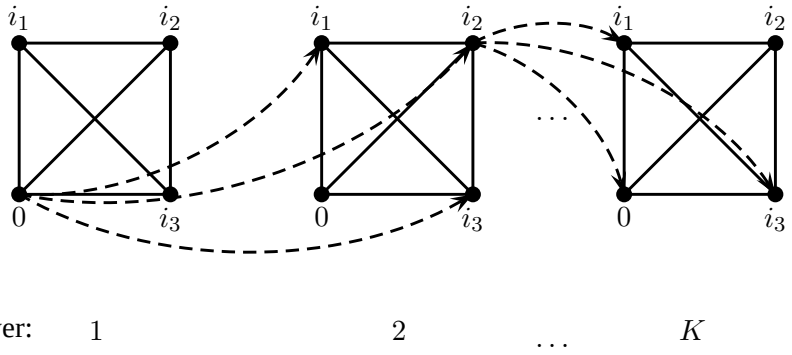



Figure 5.4: Small subset of the connector arcs. Connector arcs from node 0 in layer 1 to nodes in layer 2, and connector arcs from node 2 in layer 2 are shown as dashed lines. Not all connector arcs are shown due to readability of the graph.

nodes $0$,$i_2$,$i_3$ in layer 2, and likewise for nodes $i_2$ and $i_3$ in layer 1 has connectors to layer 2. In layer 2 the dashed lines from node $i_2$ illustrates its connectors to layer 3. Similare all other nodes in layer 2 has connectors to layer 3. In layer 3 the dashed lines illustrates the final set of connectors, which are the last edges that can be used in the system and they therefor point to the depot from all nodes. The connector arcs plays the same role as the additional arc in the pricing problem suggested above. They make it possible to obtain a path which exceeds $L-1$ by the demand of a single customer. By allowing $K$ connector arcs it is therefore possible to obtain a solution to the problem where all the $K$ layers include one additional node.

To model the connector arcs we introduce new variables $y_{ij}^k$ for all $(i,j) \in A$ and for all $k \in \mathcal{K}$. These variables substitute the variables $x_{ii}^k$ by connecting every node $(i,k) \in V^k$ in each layer $k \in \mathcal{K}$ with the nodes $(j,k+1) \in V^{k+1} : (i,j) \in A$ in the subsequent layer. Furthermore, constraints (5.11) are modified to:

$$\sum_{k \in \mathcal{K}} \sum_{(j,i) \in \delta^-(i)} \left( x_{ji}^k + y_{ji}^k \right) = \sum_{(i,j) \in \delta^+(i)} \left( x_{ij}^k + y_{ij}^k \right), \quad \forall i \in V$$

This ensures the global flow by taking the flow of the connector arcs into account. A similar substitution is made in constraint (5.12) and (5.13). The connector arcs are also present in the resource constraints where they are added to any sum bounding the resource variables. Constraint (5.15) is therefore changed to:

$$\sum_{k \in \mathcal{K}} \sum_{(j,i) \in \delta^-(i)} \left( T_{ji}^{rk} + \tau_{ji}^r \left( x_{ji}^k + y_{ji}^k \right) \right) \leq \sum_{k \in \mathcal{K}} \sum_{(i,j) \in \delta^+(i)} T_{ij}^{rk}, \quad \forall r \in R, \forall i \in C$$

A similar addition is made for constraints (5.16), (5.17), and (5.18).

When the model is decomposed into the $K$ pricing problems each set of up to $K$ connector arcs $y_{ij} : y_{ij}^k, (i,j) \in A, k \in \mathcal{K}$ becomes a single connector arc connecting the paths ending in node $i$ with the path starting in node $j$. Using the aggregated connector arcs constraints (5.23) are substituted with:

$$\sum_{p \in P: e^p = i} \lambda_p + \sum_{j \in \delta^-(i)} y_{ji} = \sum_{j \in \delta^+(i)} y_{ij} + \sum_{p \in P: s^p = i} \lambda_p \quad \forall i \in V$$

## 5.6 Resource Limit Cuts

One of the key issues with the new partial path formulation is that in a fractional solution a several routes may be exceeding the resources. The issue arises from the combination of partial paths and connectors where each connector can contribute (Make example) to an increase in the resource accumulation. The simplest form of resource limit cuts are the infeasible path inequalities (IPEC). The cuts are formulated in the original space as follows. Wlog. given a path $p = \{0, 1, 2, ..., j, 0\}$, which violates some resource $r \in \mathcal{R}$ the inequality:

$$x_{01} + x_{j0} + \sum_{i=1}^{j-1} x_{i,i+1} \leq |p| - 2$$

is valid. A key issue with IPECs is that the longer a path becomes the closer to one each edge on the path can be, while not violating the inequality. Kallehauge et al. [24] propose several ways to lift IPECs, but the issue with the cut still remains.

## 5.7 Computational Results

In this section we present the computational results for the proposed partial path model. We mainly focus on the theoretical strength of the model, and hence only report bounds at the root node. The test is divided into a comparison with algorithms for the CVRP and for the VRPTW, respectively. We have run our model on the the well-studied CVRP instances by Augerat et al. and Christofides and Eilon available at www.branchandcut.org. For the VRPTW we use the Solomon Type 1 and 2 instances with 100 customers.

For the CVRP instances we show how our bound compares to the Branch-and-Cut bound obtained by adding the capacity cuts and the Branch-and-Cut-and-Price bound obtained using the 2-cyc-SPPRC as the pricing problem. Both bounds have been reported by Fukasawa et al. [19]. For the VRPTW instances we compare our bound to the Branch-and-Cut bound computed by Kallehauge et al. [24] and the elementary bounds computed by Petersen et al. [29].

For the partial path model we add the capacity constraints using the separation algorithm by Lysgaard [27] and do not include the time variables in the model. For the CVRP instances the size of the partial paths $L$ are set to half the vehicle

| Instance | BAC | PAR | 2-cyc | OPT | Instance | BAC | PAR | 2-cyc | OPT |
|---|---|---|---|---|---|---|---|---|---|
| A-n53-k7 | 996.6 | 996.3 | 1002.2 | 1010 | B-n50-k7 | 740.0 | 741.0 | 741.0 | 741 |
| A-n54-k7 | 1130.7 | 1133.8 | 1150.3 | 1167 | B-n50-k8 | 1279.2 | 1279.8 | 1291.8 | 1312 |
| A-n55-k9 | 1055.9 | 1056.1 | 1066.4 | 1073 | B-n51-k7 | 1024.6 | 1024.6 | 1025.9 | 1032 |
| A-n60-k9 | 1316.5 | 1316.7 | 1341.6 | 1354 | B-n52-k7 | 745.0 | 745.3 | 746.4 | 747 |
| A-n61-k9 | 1004.8 | 1006.7 | 1018.8 | 1034 | B-n56-k7 | 703.4 | 703.6 | 704.5 | 707 |
| A-n62-k8 | 1244.1 | 1249.1 | 1273.2 | 1288 | B-n57-k7 | 1148.6 | 1148.6 | 1150.9 | 1153 |
| A-n63-k9 | 1572.2 | 1578.4 | 1603.5 | 1616 | B-n57-k9 | 1586.7 | 1588.8 | 1589.2 | 1598 |
| A-n63-k10 | 1262.2 | 1264.4 | 1294.2 | 1314 | B-n63-k10 | 1478.9 | 1479.5 | 1484.2 | 1496 |
| A-n64-k9 | 1340.1 | 1345.3 | 1378.8 | 1401 | B-n64-k9 | 858.5 | 859.1 | 860.2 | 861 |
| A-n65-k9 | 1151.1 | 1152.0 | 1166.6 | 1174 | B-n66-k9 | 1295.2 | 1295.8 | 1303.6 | 1316 |
| A-n69-k9 | 1108.9 | 1110.9 | 1138.7 | 1159 | B-n67-k10 | 1023.8 | 1024.0 | 1026.4 | 1032 |
| P-n50-k8 | 596.9 | 600.8 | 615.7 | 631 | B-n68-k9 | 1256.8 | 1257.0 | 1261.6 | 1272 |
| P-n55-k10 | 646.7 | 660.3 | 680.0 | 604 | B-n78-k10 | 1202.3 | 1202.4 | 1212.6 | 1221 |
| P-n55-k15 | 895.1 | 904.6 | 967.5 | 989 | E-n51-k5 | 514.5 | 514.6 | 519.0 | 521 |
| P-n60-k10 | 708.3 | 715.5 | 737.2 | 744 | E-n76-k7 | 661.4 | 663.1 | 669.9 | 682 |
| P-n60-k15 | 903.3 | 926.9 | 961.2 | 968 | E-n76-k8 | 711.2 | 714.3 | 726.0 | 735 |
| P-n65-k10 | 756.5 | 763.7 | 785.2 | 792 | E-n76-k10 | 789.5 | 796.4 | 816.8 | 830 |
| P-n70-k10 | 786.9 | 791.8 | 813.4 | 827 | E-n76-k14 | 948.1 | 964.2 | 1004.8 | 1021 |

Table 5.1: Lower bounds Results for CVRP

capacity. If there exist customers where $d_i \geq L - d_{min}$ they are removed from the total sum before $K$ is calculated. After calculation of $K$, $L$ is recalculated based on the modified total sum $Q'_t$ and $K$ such that $K = \lceil Q'_t/L \rceil$. For the VRPTW instances the capacity is varied between $\frac{1}{2}$ and $\frac{1}{15}$ of the vehicle capacity.

### 5.7.1 Results for CVRP

In table 5.1 we compare the lower bound obtained by the partial path to the bounds of Branch-and-Cut with capacity inequalities (BAC) and the Branch-and-Cut-and-Price algorithm using two cycle elimination (2-cyc). On the A, B, and E instances the bound of the partial path algorithm (PAR) is not much better than the bound of the BAC algorithm and is far from the bound of the 2-cyc algorithm. For one single instance A-n53-k7 the bound is worse. For the P instances the bound is a bit better than the BAC bound but still much worse than the 2-cyc bound. In general we can conclude that it does not appear as a good idea to pursue a Branch-and-Cut-and-Price algorithm for CVRP based on the Partial Path relaxation idea.

### 5.7.2 Results for VRPTW

The results for the VRPTW are divided into two tables. For the Type 1 Solomon instances we have not been able to find a Branch-and-Cut bound in the literature

and therefore we only report the lower bound obtained by Branch-and-Cut-and-Price with the ESPPRC as the pricing problem (ESPPRC). Furthermore, the value of $L$ can be chosen higher than for the Type 2 Solomon instances since the Type 1 Solomon instances have much smaller vehicle capacity. For the Type 2 Solomon instances the computed bounds for the partial path algorithm with a Branch-and-Cut (BAC) bound are also compare.

Results for the Type 1 Solomon instances are reported in Table 5.2. We report the lower bound found by the proposed algorithm for various capacity bounds, $f = Q/L$. As can be seen the bound does not increase much when $f$ is changed from $4$ to $3$. However, for most of the R and RC instances the bound changes a bit when moving from $f = 3$ to $f = 2$. In general, the best bound for the partial path algorithm is far from the bound of the ESPPRC algorithm on the R and RC instances.

In Table 5.3 we compare the obtained bounds for the Solomon Type 2 instances. For the C instances the bound found by the PAR algorithm is almost the same as the bound by BAC and ESPPRC. For both R and RC the bound found by the PAR algorithm is often far from the bound found by the BAC algorithm's, and even further from the bound found by the ESPPRC algorithm's. However, in the case of RC208, the best bound obtained by the PAR algorithm is better than the bound obtained by the BAC algorithm. For the Type 2 Solomon instances a small increase in the bound of the partial path algorithm is seen as $L$ increases.

| | | **ESPPRC** | **PAR** | | |
|---|---|---|---|---|---|
| **Instance** | **Opt.** | | $s = 2$ | $s = 3$ | $s = 4$ |
| R101 | 1637.7 | 1631.2 | 1624.0 | 1611.9 | 1611.9 |
| R102 | 1466.6 | 1466.6 | 1094.4 | 1071.8 | 1071.8 |
| R103 | 1208.7 | 1206.8 | 880.1 | 874.3 | 874.4 |
| R104 | 971.5 | 956.9 | 812.3 | 812.0 | 812.1 |
| R105 | 1355.3 | 1346.2 | 1204.0 | 1160.1 | 1159.0 |
| R106 | 1234.6 | 1227.0 | 943.2 | 937.5 | 937.7 |
| R107 | 1064.6 | 1053.3 | 829.9 | 830.0 | 829.8 |
| R108 | 932.1 | 913.6 | 809.1 | 808.8 | 809.0 |
| R109 | 1146.9 | 1134.3 | 884.5 | 864.1 | 864.0 |
| R110 | 1068.0 | 1055.6 | 812.9 | 812.4 | 812.4 |
| R111 | 1048.7 | 1034.8 | 822.2 | 822.1 | 821.9 |
| R112 | 948.6 | 926.8 | 804.3 | 804.3 | 804.3 |
| C101 | 827.3 | 827.3 | 827.3 | 827.3 | 827.3 |
| C102 | 827.3 | 827.3 | 819.9 | 819.9 | 820.0 |
| C103 | 826.3 | 826.3 | 819.9 | 819.9 | 820.0 |
| C104 | 822.9 | 822.9 | 818.0 | 818.0 | 818.0 |
| C105 | 827.3 | 827.3 | 827.3 | 827.3 | 827.3 |
| C106 | 827.3 | 827.3 | 827.3 | 827.3 | 827.3 |
| C107 | 827.3 | 827.3 | 827.3 | 827.3 | 827.3 |
| C108 | 827.3 | 827.3 | 818.9 | 818.8 | 818.9 |
| C109 | 827.3 | 827.3 | 817.8 | 817.8 | 817.8 |
| RC101 | 1619.8 | 1584.1 | 1324.4 | 1286.4 | 1286.4 |
| RC102 | 1457.8 | 1406.3 | 1030.2 | 1030.1 | 1030.1 |
| RC103 | 1258.0 | 1225.6 | 979 | 978.8 | 978.9 |
| RC104 | 1132.3 | 1101.9 | 968.8 | 968.5 | 968.7 |
| RC105 | 1513.7 | 1472.0 | 1097.9 | 1092.1 | 1092.1 |
| RC106 | 1401.2 | 1318.8 | 1036,2 | 1035.0 | 1034.7 |
| RC107 | 1207.8 | 1183.4 | 973.8 | 973.6 | 973.8 |
| RC108 | 1114.2 | 1073.5 | 964.1 | 964.0 | 963.6 |

Table 5.2: Lower bound results for the VRPTW for the Solomon type 1 instances. $s$ is the fraction of the original capacity of the vehicle, that is $L = \frac{Q}{s}$.

| | | **ESPPRC** | **BAC** | **PAR** | | |
|---|---|---|---|---|---|---|
| **Instance** | **Opt.** | | | $s = 8$ | $s = 10$ | $s = 15$ |
| R201 | 1143.2 | 1140.3 | 1123.6 | 1055.0 | 1040.1 | 1028.2 |
| R202 | 1029.6 | 1022.3 | 888.6 | 772.3 | 761.0 | 758.5 |
| R203 | 870.8 | 867.0 | 748.1 | 666.3 | 665.6 | 665.6 |
| R204 | - | - | 661.9 | 645.0 | 645.0 | 645.0 |
| R205 | 949.8 | 939.0 | 899.7 | 795.5 | 785.4 | 779.3 |
| R206 | 875.9 | 866.9 | 783.6 | 690.1 | 685.1 | 684.7 |
| R207 | 794.0 | 790.7 | 714.8 | 657.5 | 657.5 | 657.5 |
| R208 | - | - | 651.6 | 644.3 | 644.3 | 644.3 |
| R209 | 854.8 | 841.5 | 785.2 | 693.1 | 686.3 | 684.9 |
| R210 | 900.5 | 889.4 | 798.2 | 693.3 | 687.5 | 686.1 |
| R211 | - | - | 645.1 | 644.3 | 644.3 | 644.3 |
| C201 | 589.1 | 589.1 | 589.1 | 589.1 | 589.1 | 589.1 |
| C202 | 589.1 | 589.1 | 589.1 | 587.9 | 587.9 | 587.9 |
| C203 | 588.7 | 588.7 | 584.4 | 581.7 | 581.7 | 581.7 |
| C204 | 588.1 | 588.1 | 583.5 | 578.6 | 578.6 | 578.6 |
| C205 | 586.4 | 586.4 | 586.4 | 582.7 | 582.2 | 582.2 |
| C206 | 586.0 | 586.0 | 586.0 | 582.2 | 582.2 | 582.2 |
| C207 | 585.8 | 585.8 | 585.6 | 584.5 | 584.5 | 584.5 |
| C208 | 585.8 | 585.8 | 585.8 | 582.2 | 582.2 | 582.2 |
| RC201 | 1261.8 | 1256.0 | 1249.2 | 1121.0 | 1104.8 | 1099.4 |
| RC202 | 1092.3 | 1088.1 | 940.1 | 726.2 | 721.6 | 721.6 |
| RC203 | 923.7 | 922.6 | 781.6 | 664.6 | 664.1 | 664.1 |
| RC204 | - | - | 692.7 | 653.1 | 653.1 | 653.1 |
| RC205 | 1154.0 | 1147.7 | 1081.7 | 827.7 | 817.1 | 816.6 |
| RC206 | 1051.1 | 1038.6 | 974.8 | 816.7 | 811.0 | 811.0 |
| RC207 | 962.9 | 947.4 | 832.4 | 686.4 | 686.4 | 686.4 |
| RC208 | - | - | 647.7 | 651.7 | 651.7 | 651.7 |

Table 5.3: Lower bound results for the VRPTW for the Solomon type 2 instances. $s$ is the fraction of the original capacity of the vehicle, that is $L = \frac{Q}{s}$.

## 5.8 Conclusion and Future Work

A new decomposition model of the VRP has been presented with the ESPPRC as the pricing problem. The model makes it possible to balance the running time of the pricing problem against the tightness of the lower bound. Due to the aggregation of the model, LP relaxed bounds of (5.21)–(5.28) are better than the direct model (5.1)–(5.7). Since (5.21)–(5.28) is a generalization of the traditional Dantzig-Wolfe decomposition model with elementary routes as columns, the LP relaxed bounds may be both weaker and stronger. It has been shown that the bound of the presented LP relaxation is sometimes better than that of the classical decomposition of VRP into an elementary route for each vehicle.

**Future work:** The quality of the bounds can be further improved by using special purpose cutting planes, which this paper has not focused on. Furthermore, effective cuts such as Subset Row-inequalities by Jepsen et al. [23] and Chvátal-Gomory Rank-1 cuts (see Petersen et al. [29]) can be applied to the Set Partition master problem to strengthen the bound.

More and better cuts have been added to the VRPTW Branch-and-Cut algorithm used in this paper for comparison, but all of these cuts could also be added to this model obtaining at least as good a bound.

Considering the approach of Baldacci et al. [2] where columns are enumerated dependent on strong upper and lower bounds, it should be clear that the partial path approach should contain fewer enumerated columns due to the smaller solution space of the pricing problem. Combining the relatively strong bound with the small solution space a powerful strategy should be obtained.

# Bibliography


[1] N. Ascheuer, M. Fischetti, and M. Grötschel. Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. *Mathematical Programming*, 90(3):475–506, 2001.

[2] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008.

[3] J. F. Bard, G. Kontoravdis, and G. Yu. A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36(2): 250–269, 2002.

[4] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1998.

[5] J.E. Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19:379–394, 1989.

[6] N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Oper. Res. Lett.*, 34(1):58–68, 2006.

[7] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 23(10):2972–2990, 2005.

[8] N. Christofides and S. Eilon. An algorithm for the vehicle-dispatching problem. *Operational Research Quarterly*, 20(3):309–318, 1969.

[9] N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. *Mathematical Programming*, 20(1):255–282, 1981.

[10] W. Cook and J. L. Rich. A parallel cutting plane algorithm for the vehicle routing problem with time windows. Technical Report TR99-04, Computational and Applied Mathematics, Rice University, Houston, Texas, USA, 1999.

[11] E. Danna and C. Le Pape. Accelerating branch-and-price with local search: A case study on the vehicle routing problem with time windows. In G. Desaulniers, J. Desrosiers, , and M. M. Solomon, editors, *Column Generation*, chapter 3, pages 30–130. Springer, 2005.

[12] G. Desaulniers, J. Desrosiers, J. Ioachim, I. M. Solomon, F. Soumis, and D. Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T. G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, chapter 3, pages 57–93. Springer, 1998.

[13] G. Desaulniers, F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. *Transportation Science*, 42(3):387–404, 2008.

[14] M. Desrochers. *La fabrication d'horaires de travail pour les conducteurs d'autobus par une méthode de génération de colonnes*. PhD thesis, Université de Montréal, Canada, 1986.

[15] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

[16] M. Dror. Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, 42:977–979, 1994.

[17] I. Dumitrescu. *Constrained Path and Cycle Problems*. PhD thesis, Department of Mathematics and Statistics, University of Melbourne, Australia, 2002.

[18] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

[19] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R.F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming*, 106(3):491–511, 2006.

[20] S. Irnich. Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectrum*, 30(1):113–148, 2008.

[21] S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, Jacques Desrosiers, and M.M. Solomon, editors, *Column Generation*, chapter 2, pages 33–65. Springer, 2005.

[22] S. Irnich and D. Villeneuve. The shortest-path problem with resource constraints and k-cycle elimination for k $\geq$ 3. *INFORMS J. on Computing*, 18 (3):391–406, 2006.

[23] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle routing problem with time windows. *Operations Research*, 56(2):497–511, 2008.

[24] Brian Kallehauge, Natashia Boland, and Oli B. G. Madsen. Path inequalities for the vehicle routing problem with time windows. *Networks*, 49(4):273–293, 2007. ISSN 0028-3045.

[25] N. Kohl, J. Desrosiers, O. B. G. Madsen, M. M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999.

[26] M. E. Lubbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[27] J. Lysgaard. CVRPSEP: A package of separation routines for the capacitated vehicle routing problem. Available at: url: www.asb.dk/∼lys, 2003.

[28] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle problem. *Mathematical Programming*, 100 (A):423–445, 2004.

[29] B. Petersen, D. Pisinger, and S. Spoorendonk. Chvátal-Gomory rank-1 cuts used in a Dantzig-Wolfe decomposition of the vehicle routing problem with time windows. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 397–420. Springer, 2008.

[30] G. Righini and M. Salani. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006.

[31] G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Netw.*, 51(3):155–170, 2008. ISSN 0028-3045.

[32] M. Salani. *Branch-and-Price Algorithms for Vehicle Routing Problems*. PhD thesis, Universitá Degli Studi Di Milano, Facoltá di Scienza Matematiche, Fisuche e Naturali Dipartimento di Technologie dell'Informazione, Milano, Italy, 2005.

[33] M. M. Solomon. Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research*, 35:234–265, 1987.

[34] P. Toth and D. Vigo. *The Vehicle Routing Problem*, volume 9 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 2001.

[35] F. Vanderbeck. On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Operations Research*, 48(1):111–128, 2000.

# Chapter 6 Partial Path Column Generation for the Elementary Shortest Path Problem with a Capacity Constraints

Mads Jepsen Bjørn Petersen
*DTU Managment Engineering*

June 2009

## Abstract

This paper introduces a decomposition of the Elementary Shortest Path Problem with a Capacity Constraints(ESPPCC), where the path is combined by smaller sub paths. We show computationals result by comparing different approaches for the decomposition and compare the best of these with existing algorithms. We show that the algorithm for many instances outperforms a bidirectional labeling algorithm.

## 6.1 Introduction

A formal definition of the ESPPCC problem solved in this paper is as follows: We are given a directed $G(V_d,A)$ with node set $V = \{o,v_1,..,d\}$, an arc set $A$ and a resource with a global upper bound $W$. The nodes $o$ and $d$ are refered to as the origin and destination. For each arc $(i,j) \in A, c_{ij}$ is the cost of the arc and for each node $i \in V$, $w_i$ is the resource consumption of the node. A path $p$ is an ordered vertex set which is given as $V(p)$ and an ordered arc set given as $A(p)$. A path is feasible if $\sum_{(i)\in V(p)} w_i \leq W$ and it starts in $o$ and ends in $d$. The objective is to find the feasible path $p$ which minimizes the cost $\sum_{(i,j)\in A(P)} c_{ij}$.

ESPPCC is a special case of the more general problem called ESPPRC which may contain more than one resource and where resource can also be locally constrained. When negative cycles are allowed, the ESPPRC can be shown to be NP-complete by reduction from the longest path problem. Beasley and Christofides [2] gave a mathematical formulation of the problem where each node is considered a resource. When the graph may contain negative cost cycles Feillet et al. [9] introduced a labeling, Righini and Salani [14] proposed a bi-directional labeling algorithm and a Branch and Bound algorithm, using a relaxation where cycles are allowed [1] , Boland et al. [3] gave a label correcting algorithm and Baldacci et al.

[1] see Irnich and Villeneuve [11] for details on the relaxation

[1] computed lower bounds on paths from a node in the graph to the destination and used these to speed up a bi-directional labeling algorithm. The ESPPCC has a structure similar to the profitable tour problem named by Dell'Amico et al. [7], a problem that falls within the category of traveling salesman problems with profits as classified by Feillet et al. [10].

For ESPPRC where the graph is assumed to contain no negative cost cycles, known as the resource constraint shortest path problem (SPPRC), Beasley and Christofides [2] gave a Branch and Bound algorithm based on Lagrangian dual bounds, Dumitrescu and Boland [8] suggest improved prepossessing as well as several algorithms and Carlyle et al. [5] propose a Lagrangian approach, where paths with cost between the Lagrangian bounds and the current upper bound are found using the k shortest path algorithm by Carlyle and Wood [4]

The main application of ESPPRC is as the pricing problem, when solving the Vehicle Routing Problem through Branch and Cut and Price. Chabrier [6] and Jepsen et al. [13] has done this successfully for VRPTW and Baldacci et al. [1] has done it for CVRP.

Labeling algorithms has so far been used very successfully for ESPPRC problems especially when time windows are included as resources. However for instances where the time windows are very large the state space becomes huge and labeling algorithms are no longer a good practical solution approach.

Motivated by the bi-directional labeling algorithm by Righini and Salani [14] and the fact that branch and cut has been used quite successfully to solve the ESPPRC when time window like resources are not included(see Jepsen et al. [12]), we propose a Danzig-Wolfe decomposition approach based on a model where small sub paths called partial paths are concatenated together to form the solution. Since each of the sub paths are elementary the SR-inequalities for VRPTW introduced by Jepsen et al. [13] can be used to improve the lower bound. Furthermore any valid inequality to the ESPPRC can be used.

## 6.2 Bounded partial paths

The idea behind the following mathematical model and decomposition is that any feasible path $p$ can be seen as a sequence of $K = \{1,..,|K|\}$ partial paths $p_{ov_1}, p_{v_1v_2}, ...., p_{v_kd}$. Where $p_{ij}$ is a partial path from node $i$ to node $j$.

Each of the $|K|$ partial paths can be seen as a path through the original graph. This leads to a formulation of the ESPPRC where the graph is replicated $|K|$ times and arcs are added between the adjacent replications. We shall refer to these replications as layers. To ensure a correct division of any feasible path, let $w_{\max}$ be the maximal resource consumption. For a fixed number of partial paths $|K|$, the maximal partial path length $L$ is given as:

$$L = \left\lceil \frac{W}{|K|} \right\rceil + w_{\max} - 1$$

The idea is to use $L$ to bound the length of the partial paths, the ideal value would be as close as possible to $\frac{W}{|K|}$. But since we need a fixed value it is necessary to add the maximal resource consumption to ensure the feasibility.

Let $\delta^+(S) = \{(i,j) \in A | i \in S\}$ denote arcs out of the set $S$ and $\delta^-(S) = \{(i,j) \in A | j \in S\}$ denote the in going arcs of $S$. We shall use $\delta(i)$ instead of $\delta(\{i\})$ for $i \in V$. The binary variable $x_{ijk}$ is 1 if arc $(i,j) \in A$ is used in the $k$'th layer and 0 otherwise. The binary variables $s_{ik}$ indicates if a partial path starts in node $i \in V$ in layer $k \in K$ and the binary variables $t_{ik}$ indicates if a partial path ends in node $i \in V$ in layer $k$. For ease of modelling we assume that $t_{i|K|+1} = s_{i0} = 0, \ \forall i \in V$. The mathematical model for ESPPCC based on partial paths can be formulated as:

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ijk} \tag{6.1}$$

$$\texttt{s.t.} \sum_{(o,j) \in \delta^+(o)} x_{oj1} = 1 \tag{6.2}$$

$$\sum_{(i,d) \in \delta^-(S)} x_{id|K|} = 1 \tag{6.3}$$

$$\sum_{k \in K} \sum_{(i,j) \in A} x_{ijk} \leq 1 \qquad \forall\, v \in V \setminus \{o,d\} \tag{6.4}$$

$$\sum_{k \in K} \sum_{(i,j) \in A} w_i x_{kij} \leq W \tag{6.5}$$

$$\sum_{k \in K} \sum_{(i,j) \in \delta+(S)} x_{ijk} \geq \sum_{k \in K} \sum_{(i,j) \in \delta^+(s)} x_{ijk} \qquad S \subseteq V, \forall s \in S \tag{6.6}$$

$$\sum_{i \in V} s_{ik} = 1 \qquad \forall k \in K \tag{6.7}$$

$$t_{ik-1} = s_{ik} \qquad \forall i \in V, \forall k \in K \tag{6.8}$$

$$s_{ok} = \sum_{(i,j) \in \delta^+(i)} x_{ij1} \qquad \forall k \in K \tag{6.9}$$

$$\sum_{(j,i) \in \delta^-(i)} x_{jik} = t_{dk} \qquad \forall k \in K \tag{6.10}$$

$$s_{ik} + \sum_{(j,i) \in \delta^-(i)} x_{jik} = t_{ik} + \sum_{(i,j) \in \delta^+(i)} x_{ijk} \qquad \forall i \in V \setminus \{o,d\}, k \in K \tag{6.11}$$

$$\sum_{(i,j) \in \delta+(S)} x_{ijk} \geq \sum_{(i,j) \in \delta^+(s)} x_{ijk} \qquad \forall k \in K, S \subseteq V, \forall s \in S \tag{6.12}$$

$$\sum_{(i,j) \in A} w_i x_{ijk} \leq L \qquad \forall k \in K \tag{6.13}$$

$$x_{ijk} \in \{0,1\} \qquad \forall (i,j) \in A, k \in K \tag{6.14}$$

$$t_{ik}, s_{ik} \in \{0,1\} \qquad \forall\, i \in V, k \in K \tag{6.15}$$

The objective (6.1) is to minimize the total cost of the path. Constraints (6.2), (6.3) and (6.4) ensure that no node is visited more than once and that the path starts in the source node in the first layer and finises in the target node in the final layer. Constraint (6.5) is the resource bound and constraints (6.6) are the generalized

subtour inequalities(GSEC) which prevents cycles in a solution. Constraints (6.7) ensure that a partial path starts in each layer and constraints 6.8 ensure that if a partial path ends in a node in the previous layer it starts in that node in the current layer. Constraints (6.9) and (6.9) ensure that a partial path can only start in origin and end in the destination. Constraints (6.11) are flow conservation constraints for each of the layers, constraints (6.12) eliminate subtours within each layer and constraints (6.13) are bounds on each of the partial paths. Finally the domains of the variables are defined in constraints (6.14). It is worth noting that even though it is enforced that $|K|$ partial paths are used, it is possible to make an empty partial path in a layer by setting $t_{ik} = s_{ik}$.

In the following we will make a Danzig-Wolfe reformulation of the mathematical model, where constraints 6.11 to 6.13 form $K$ identical sub problems. This can be rewritten to a single sub-problem where the goal is to find a resource constrained shortest path $p$ between two arbitrary nodes in the graph. Let $\alpha_{ij}^p = 1$ if path $p$ use arc $(i, j)$ and zero otherwise, $\beta_i^p = 1$ if $p$ starts in node $i$ and $\gamma_i^p$ indicate if $p$ ends in node $i$ . The binary variable $\lambda^p$ indicate if partial path $p$ is used and $c_p$ be the cost of using the path. The master problem then becomes:

$$\min \sum_{p \in P} c_p \lambda_p \tag{6.16}$$

$$\texttt{s.t.} \sum_{p \in P} \sum_{(o,j) \in \delta^+(o)} \alpha_{oj}^p \lambda_p = 1 \tag{6.17}$$

$$\sum_{p \in P} \sum_{(i,d) \in \delta^-(S)} \alpha_{id}^p \lambda_p = 1 \tag{6.18}$$

$$\sum_{p \in P} \sum_{(i,j) \in A} \alpha_{ij}^p \lambda_p \leq 1 \qquad \forall\, v \in V \setminus \{o,d\} \tag{6.19}$$

$$\sum_{p \in P} \sum_{(i,j) \in \delta+(S)} \alpha_{ij}^p \lambda_p \geq \sum_{p \in P} \sum_{(i,j) \in \delta^+(s)} \alpha_{ij}^p \lambda_p \qquad S \subseteq V, \forall s \in S \tag{6.20}$$

$$\sum_{p \in P} \sum_{(i,j) \in A} w_{ij} \alpha_{ij}^p \lambda_p \leq W \tag{6.21}$$

$$\sum_{p \in P} \lambda^p \leq |K| \tag{6.22}$$

$$\sum_{p \in P} \gamma_i^p \lambda_p = \sum_{p \in P} \beta_i^p \lambda_p \qquad \forall i \in V \setminus \{o,d\} \tag{6.23}$$

$$\lambda_p \in \{0,1\} \qquad \forall\, p \in P \tag{6.24}$$

With the exception of constraint 6.22 the constraints follow directly from a standard Danzig-Wolfe reformulation. The objective (6.16) minimizes the cost of the selected partial paths. Constraints (6.17) and (6.18) ensure that the origin is left and the destination is entered by a partial path. Constraints (6.19) ensure that the nodes in the graph are visited at most once. Constraints (6.20) are the generalized subtour elimination constraints and constraint (6.21) ensure that the global resource is not exceeded. Constraints (6.22) are the convexity constraints which ensure that at most $|K|$ partial paths are selected. It is a less than or equal constraint to avoid the

need of empty partial paths. Constraints (6.23) ensure that if a partial path ends in a node different from the destination, an other partial path will start in that node. Finally the domains of the variables are defined in (6.24). To use Branch-and-Cut-and-Price the domains are relaxed to $0 \leq \lambda_p \leq 1 \; \forall p \in P$ and columns are generated by solving a pricing problem.

Let $\pi_i$ be the $|V|$ dual of constraints (6.17), (6.18) and (6.19), $\sigma$ be the dual of constraints (6.21) and $\rho_i$ be the $|V|$ duals of constraints (6.23). To calculate the reduced cost of a column in the master problem, we set the edge cost to:

$$\hat{c}_{ij} = \left\{ \begin{array}{l} c_{id} - \pi_i - pi_d \forall i \in V \setminus \{o,d\} \\ c_{ij} - \pi_i - w_i \sigma \quad \forall i \in V \setminus \{d\} \end{array} \right.$$

Let $x_{ij}$ be the binary variable which defines if arc $(i,j) \in A$ is used, the binary variable $ls_i$ indicate if the path starts in node $i \in V$ and the binary variable $lt_i$ indicate if the path ends in node $i \in V$. The mathematical model for the pricing problem then becomes:

$$\min \sum_{(i,j)\in A} \hat{c}_{ij} x_{ij} + \sum_{i \in V \setminus \{o,d\}} (\rho_i ls_i - \rho_i lt_i) \tag{6.25}$$

$$\sum_{i \in V} ls_i \leq 1 \tag{6.26}$$

$$\sum_{i \in V} lt_i \leq 1 \tag{6.27}$$

$$ls_o = \sum_{(i,j)\in \delta^+(i)} x_{ij} \tag{6.28}$$

$$\sum_{(j,i)\in \delta^-(i)} x_{ji} = t_d \tag{6.29}$$

$$ls_i + \sum_{(j,i)\in \delta^-(i)} x_{ji} = lt_i + \sum_{(i,j)\in \delta^+(i)} x_{ij} \qquad \forall i \in V \setminus \{o,d\} \tag{6.30}$$

$$\sum_{(i,j)\in \delta+(S)} x_{ij} \geq \sum_{(i,j)\in \delta^+(s)} x_{ij} \qquad S \subseteq V, \forall s \in S \tag{6.31}$$

$$\sum_{(i,j)\in A} w_i x_{ij} \leq L \tag{6.32}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{6.33}$$

$$ls_i, lt_i \in \{0,1\} \qquad \forall \, i \in V \tag{6.34}$$

The objective of the pricing problem minimizes the cost of the selected partial path. Constraints (6.26) to (6.29) ensure that only one starting and ending variable is used and that the a path may only start in the origin and end in the destination. Constraints (6.30) to (6.31) ensure that the partial path is simple and connected. Constraint (6.32) ensures that the consumption of the resource is no higher than $L$. Finally the domains are defined by constraints (6.33) .

A column has negative reduced cost if it is less than the dual variable of constraint 6.22.

To solve the pricing problem we reformulate it to an ESPPRC. This is done by substituting the variables $ls_i$ and $lt_i$ with arcs from a fake source node and arcs to a fake target node.

More formally we define a fake source node $\bar{s}$ and a fake target node $\bar{t}$. The fake arc set $\bar{A} = \{(\bar{s}, v) : v \in V\} \cup \{(v, \bar{s}) : v \in V\}$. The pricing problem then becomes solving an ESPPRC with a single resource in the graph $\bar{G}(V \cup \bar{V}, A \cup \bar{A})$ where the cost is $\bar{c}_{ij} = \hat{c}_{ij}(i,j) \in A$ and $\bar{c}_{ij} = 0, (i,j) \in \bar{A}$

The lower bound can be improved using valid inequalities for the ESPPRC polytope see Chapter 4 for details. Valid inequalities for the master model such as the SR-inequalities by Jepsen et al. [13] and Strong Capacity Constraints by Baldacci et al. [1] can also be adapted.

### 6.2.1 Using Length as a Resource

Since it is possible to determine an upper bound on the length on a path, we can use this to make an alternative division of a path. Let $W_L$ denote the maximal length of any feasible path. Then any feasible path can be divided into $|K|$ segments with at most $L = \left\lceil \frac{W_L}{|K|} \right\rceil$ nodes in each segment.

The lower bound can be improved by imposing a minimum length on each of the partial paths. Let $L_{min}$ be the minimum length of a partial path. If $2L_{min} - 1 = L$ any feasible solution with length greater than $L_{min}$ can be constructed. Solution with a length less than $L_{min}$ can be found using standard solution techniques for solving the ESPPRC.

## 6.3 Implementation

We have implemented the bidirectional labeling algorithm by Righini and Salani [14]. When we impose a lower bound on the length of the partial path we only use dominance when the labels have equal length. In the case where the capacity resource is used, there may exist nodes $i \in V \setminus \{o,d\}$, where $w_i + w_j \geq L \forall j \in V \setminus \{o,d,i\}$ in this case we add the element as a partial path a priory.

The Branch-Cut-And-Price algorithm is implemented in the BCP project from coin-or.org. We use CLP as our LP solver and we separate the GSECs solving a minimum cut problem(see Wolsey [15]) for details. The SR inequalities are separated using the algorithm proposed by Jepsen et al. [13], either the first or the last node on a partial path is not considered part of the SR-cut. Branching is done on a single arc or all arcs out of a node and is added as a cut in the master model. The constraints in the original space are:

$$\sum_{k \in K} \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 0 \qquad \sum_{k \in K} \sum_{(i,j) \in \delta^+(i)} x_{ijk} = 1 \qquad i \in V \tag{6.35}$$

$$\sum_{k \in K} x_{ijk} = 0 \qquad \sum_{k \in K} x_{ijk} = 1 \qquad (i,j) \in A \tag{6.36}$$

The decomposed version of the branches are:

$$\sum_{p\in P}\sum_{(i,j)\in\delta^+(i)}\alpha_{ij}^p\lambda_p = 0 \qquad \sum_{p\in P}\sum_{(i,j)\in\delta^+(i)}\alpha_{ij}^p\lambda_p \qquad i\in V \tag{6.37}$$

$$\sum_{p\in P}\alpha_{ij}^p\lambda_p = 0 \qquad \sum_{p\in P}\alpha_{ij}^p\lambda_p = 1 \qquad (i,j)\in A \tag{6.38}$$

## 6.4 Computational studies

Based on a column generation algorithm for CVRP, some instances for the ESPPRC were generated based on CVRP instances from www.branchandcut.org. For the general model we have tested several settings on several instances with 20-30 nodes and have based on these chosen some settings to use on larger instances.

For the instances generated we have bounded the partial path using both length and capacity. When using length we have chosen to restrict the maximal lower limit to 3 and the maximal upper limit to 5. For the capacity we split the path in pieces of at most a tenth of the total capacity, finally we included the SR inequalities for the different settings.

We have chosen to show the result for a single instance which was quit representative for the instances we benchmarked on. Furthermore the instance have the characteristics we are targeting to solve. The instance has 30 nodes, the maximal feasible path length is 23 and the capacity resource is 4500.

In table 6.1 we have compared some different settings for capacity and length. $L_{\min}$ is the minimal value of the partial path, $L_{\max}$ is the maximal value of the partial path. RB is the root bound and $T$ is the time without SR inequalities. $RB_{SR}$ and $T_{SR}$ is the root bound and time when SR inequalities is included.

| Instance | Bounded on | $L_{\min}$ | $L_{\max}$ | RB | T | $RB_{SR}$ | $T_{SR}$ |
|---|---|---|---|---|---|---|---|
| E-n30-k3-20 | Capacity | 0 | 2125 | -192.350 | 311.895 | -192.350 | 386.072 |
| E-n30-k3-20 | Capacity | 0 | 1700 | -192.350 | 228.958 | -192.350 | 203.585 |
| E-n30-k3-20 | Capacity | 0 | 1300 | -192.320 | 49.579 | -192.320 | 143.685 |
| E-n30-k3-20 | Capacity | 0 | 1193 | -192.350 | 31.810 | -192.350 | 128.412 |
| E-n30-k3-20 | Capacity | 0 | 1113 | -192.350 | 23.653 | -192.350 | 120.776 |
| E-n30-k3-20 | Capacity | 0 | 1000 | -192.350 | 39.098 | -192.350 | 171.571 |
| E-n30-k3-20 | Capacity | 0 | 900 | -192.350 | 20.173 | -192.350 | 118.271 |
| E-n30-k3-20 | Length | 0 | 3 | -192.350 | 20.361 | -192.350 | 19.893 |
| E-n30-k3-20 | Length | 0 | 4 | -192.350 | 44.407 | -192.350 | 50.455 |
| E-n30-k3-20 | Length | 0 | 5 | -192.350 | 134.080 | -192.350 | 99.106 |
| E-n30-k3-20 | Length | 0 | 6 | -192.350 | 255.236 | -192.350 | 269.989 |
| E-n30-k3-20 | Length | 2 | 3 | -192.350 | 75.873 | -192.350 | 109.283 |
| E-n30-k3-20 | Length | 2 | 4 | -192.350 | 102.402 | -192.350 | 113.751 |
| E-n30-k3-20 | Length | 3 | 5 | -192.350 | 163.822 | -192.320 | 431.763 |

Table 6.1: Comparing different schemes

From the result in table 6.1 it is clear that the longer the path the poorer the algorithms perform. The main reason for this is that no matter how long the path becomes there is simply no gain in the quality of the relaxation. The value of the root bound is almost the same as the one for branch and cut, which is $-192352.787$. When including the SR inequalities only a few of the setting results in a improvement of the running time. When a lower bound on the path length is included the running time increases in all cases and the root bound is still the same.

In table 6.2 we have shown the solution times for the two best general partial paths algorithms. $T_{len}$ is the running time of the best length algorithm and $T_{cap}$ is the running time of the best with capacity. The values are compared to the bi-directional labeling algorithm($T_{label}$) and the Branch and Cut algorithm($T_{BAC}$) by Jepsen et al. [12].

| Instance | $T_{BAC}$ | $T_{label}$ | $T_{len}$ | $T_{cap}$ |
|---|---|---|---|---|
| E-n30-k3-20 | 0.44 | $> 1800$ | 19.893 | 20.173 |
| B-n31-k5-17 | 2.07 | 0.22 | 124.492 | 24.178 |
| A-n32-k5-120 | 0.51 | 0.28 | 32.714 | 7.892 |
| A-n33-k5-31 | 0.45 | 0.01 | 121.440 | 14.477 |
| B-n34-k5-17 | 2.21 | 72.79 | 290.022 | 32.554 |
| B-n45-k6-54 | 4.63 | 90.3 | 286.978 | 109.011 |
| P-n45-k5-150 | 0.58 | 0.71 | 19.753 | 15.457 |
| P-n50-k8-19 | 0.94 | $> 1800$ | 188.008 | 25.350 |
| E-n51-k5-29 | 2.46 | $> 1800$ | 277.645 | 287.746 |

Table 6.2: Characteristics of the benchmark instances

The main conclusion when comparing the results in table 6.2 is that the branch and cut algorithm outperforms the other algorithms. The second observation is that the partial path algorithms is able to solve all instances within 30 minutes which labeling is not. It is also worth noting that the algorithm which bounds using capacity in almost all cases is considerable better than the one that bounds using length. Finally we conclude that the general partial path algorithms can not compete with the Branch And Cut algorithm.

## 6.5 Conclusion and future research

In this conference paper we have introduced an alternative formulation of ESPPRC and shown how it can be solved using the Danzig-Wolfe decomposition principle. We have shown that a early prototype is better than a standard labeling algorithm, but we have not been able to show that the bound obtained is better than a standard Branch and Cut algorithm.

# Bibliography


[1] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 2008. doi: 10.1007/s10107-007-0178-5.

[2] J.E. Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19:379–394, 1989.

[3] N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Oper. Res. Lett.*, 34(1):58–68, 2006.

[4] W. Matthew Carlyle and R. Kevin Wood. Near-shortest and k-shortest simple paths. *Netw.*, 46(2):98–109, 2005. ISSN 0028-3045. doi: http://dx.doi.org/10.1002/net.v46:2.

[5] W. Matthew Carlyle, Johannes O. Royset, and R. Kevin Wood. Lagrangian relaxation and enumeration for solving constrained shortest-path problems. *Networks*, 2008.

[6] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 23(10):2972–2990, 2005.

[7] M. Dell'Amico, F. Maffioli, and P. Värbrand. On prize-collecting tours and the asymmetric travelling salesman problem. *International Transactions in Operations Research*, 2(3):297–308, 1995.

[8] I. Dumitrescu and N. Boland. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks*, 42 (3):135–153, 2003.

[9] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

[10] D. Feillet, P. Dejax, and M. Genderau. Traveling salesman problems with profits. *Transportion Science*, 39(2):188–205, 2005.

[11] S. Irnich and D. Villeneuve. The shortest path problem with resource constraints and $k$-cycle elimination for $k \geq 3$. *INFORMS Journal on Computing*, 18:391–406, 2006. doi: 10.1287/ijoc.1040.0117.

[12] M. Jepsen, B. Petersen, and S. Spoorendonk. A branch-and-cut algorithm for the elementary shortest path problem with a capacity constraint. Technical report, DIKU, University of Copenhagen, Denmark, 2008.

[13] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle routing problem with time windows. *Operations Research*, 56(2):497–511, March–April 2008.

[14] G. Righini and M. Salani. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006.

[15] L. A. Wolsey. *Integer Programming*. John Wiley & Sons, Inc., 1998.

# Chapter 7

# Conclusion

The thesis has studied two exact methods Branch-and-Cut and Branch-and-Cut-and-Price in the context of the vehicle routing problem. The study has to this date led to one accepted journal paper, one conditionally accepted journal paper and three accepted conference papers. Two of the three conference papers have been extended and further work will be conducted on one of these. Furthermore the work carried out during the thesis has led to two full papers that are submitted for publication. Finally the thesis contains a technical report.

## 7.1 Summary

The main contributions of the thesis are:

- Using known solutions approaches on new problems
  - The adaption of the two index formulation from the Capacitated Location Routing Problem to the Two Echelon Vehicle Routing Problem.
  - The adaption of the Multistar inequalities to both the Two Echelon Vehicle Routing Problem and the Capacitated Profitable Tour Problem.
  - The adaption and proof of validity of cuts from many polytopes to the Capacitated Profitable Tour Problem
- Improvement of existing methods:
  - The development of SR-inequalities for the Branch-and-Cut-and-Price method for the Vehicle Routing Problem with Time Windows.
  - The adaption and further development of the Branch-and-Cut algorithms for the Capacitated Profitable Tour Problem. In particular the introduction of the rounded multistar inequalities.

- Development of new solution methods:
  - The development of the heuristic separation routine for the Knapsack Large Multistar inequalities in the context of the Capacitated Profitable Tour Problem.
  - The development of the Partial Path Solution framework for the Capacitated Vehicle Routing Problem, The Vehicle Routing Problem with Time Windows and the Elementary Resource Constrained Shortest Path Problem.

The results of each of the chapters in Part can be summarized as: I

**Chapter 2: Subset-Row Inequalities Applied to the Vehicle Routing Problem with Time Windows** The paper presented a BCP algorithm for the Vehicle Routing Problem with Time Windows which was able to solve 8 previously unsolved instances from the Solomon benchmarks for this problem. The two core features of the new BCP algorithm was the addition of the SR inequalities in the master problem and the handling of these in the pricing problem. the SR-inequalities have later been used by Desaulniers et al. [22] for the VRPTWand [2] for CVRP and VRPTW.

**Chapter 3: A Branch-and-Cut Algorithm for the Symmetric Two-echelon Capacitated Vehicle Routing Problem** The paper presented a BAC algorithm for the Two-Echelon Capacitated Vehicle Routing Problem, the paper introduces a two index model that is used to compute lower bounds and the computational results show that the method outperforms previous solution methods.

**Chapter 4: A Branch-and-Cut Algorithm for the Capacitated Profitable Tour Problem** The paper presented a BAC algorithm for the Capacitated Profitable Tour Problem. The paper introduces several new valid inequalities and include proof of the validity for all the adapted inequalities. The computational results show that the method can solve instances with up to 800 nodes to optimality and that the method outperform existing algorithms for large instances.

**Chapter 5: Partial Path Column Generation for the Vehicle Routing Problem** An new BCP based solution approach for the CVRP and VRPTW Problem and . In theory the new BCP algorithm seemed promising, but unfortunately the computational results indicate that this approach has some serious issues.

**Chapter 6 Partial Path Column Generation for the Elementary Shortest Path Problem with a Capacity Constraints** The paper used the Partial Path Column Generation approach to solve the Elementary Shortest Path Problem with a Capacity Constraint( This problem is very similar to Capacitated Profitable Tour Problem). The computational results indicated that a small improvement in the lower bounds could be obtained, but that the running times could not compete with the running times of the BAC algorithm developed in Chapter 4.

## 7.2 Future Research

In this section some future research ideas for each of the chapters in this thesis will be presented.

**Chapter 2 Subset-Row Inequalities Applied to the Vehicle Routing Problem with Time Windows** The SR-inequalities are already widely used in many routing problems and some extensions of these have been presented in the literature. One area where future research could be directed is within the solution of the pricing problem. As mentioned in section 1.1.3 modern ESPPRC algorithm use bounding functions. One drawback of the current bounding methods is that they do not incorporate the penalty from the SR inequalities. The perspective of a better integration of the dual penalties associated with the SR inequalities are that the solution of the pricing problem would be faster which could lead to the solution of bigger VRP instances.

**Chapter 3 A Branch-and-Cut Algorithm for the Symmetric Two-echelon Capacitated Vehicle Routing Problem** In the context of the Capacitated Location Routing Problem there has been suggested many new cutting planes. Many of these has the potential to be adapted to the 2ECVRP. Furthermore it would be interesting to integrate the model with a column generation approach in a BCP algorithm.

**Chapter 4 A Branch-and-Cut Algorithm for the Capacitated Profitable Tour Problem** The algorithm used to solve the Capacitated Profitable Tour Problem could be used to solve the pricing problem of CVRP. To do this it is necessary to handle the constraints inferred by the SR-inequalities and the strong capacity constraints. It would furthermore be necessary to develop valid inequality based on the lower bound on the capacity, since these are naturally imposed when solving the problem in a column generation context.

**Chapter 5 Partial Path Column Generation for the Vehicle Routing Problem** The huge issue with the partial path solution approach for CVRP is that for each partial path an additional edge was allowed to ensure feasibility. An idea could therefore be to transfer the solution method to a problem where this issue can be avoided. Such a problem could be the Vehicle Routing Problem with Backhauls, where the edge connecting the Linehaul and Backhaul does not have any capacity.

**Chapter 6 Partial Path Column Generation for the Elementary Shortest Path Problem with a Capacity Constraints** For the new BCP algorithm for solving the ESPPCC an issues is that branching was difficult and the solution of the LPs are slow. An idea could be to solve the master problem using an subgradient algorithm and to enumerate the columns needed when the root node is solved. Furthermore it is natural to use all the cuts developed in Chapter 4.

## 7.3 Final Remarks

Some of the work of this thesis has already been used by other authors. This is mainly the work of the SR-inequalities which has been integrated into the BCP algorithms of Desaulniers et al. [22] and Baldacci et al. [2]. In both cases the results are better than the ones presented in chapter 2. Common for both of the solution methods the SR-inequalities were included and contributed to the success of the algorithms. Whether or not more of the ideas in this thesis will be used only time will tell.

As a researcher within the field of CVRP and VRPTW I started almost a decade ago as an undergraduate student, back then one of the Solomon-instances with the 25 customers for VRPTW was still unsolved and many variants of the Vehicle Routing Problem were too complex to solve to optimality. Today there only exist one unsolved Solomon instance with 100 customers, optimal solution approaches for many variants of the Vehicle Routing Problem exist and it is not impossible that within the next decade only one 1000 customer instance of the Gehring & Homberger instances for VRPTW will remain unsolved. To achieve this goal I believe we will need to invent new cutting planes based on the master formulation that possesses some properties that make it suitable to incorporate in the bounding methods of the pricing problem. I also strongly believe that the idea of cutting on a given transaction in the full capacity state space of the CVRP(This could also be done for VRPTW) which has been presented at various conference by *Marcus Poggi de Aragão* and *Eduardo Uchoa* will find its way into the new algorithms. Finally someone will get a new brilliant idea that none of us had thought about and it will push the algorithms to the next level.

# Part II

# Appendix

# Appendix A The vehicle routing problem with edge set costs

Line Blander Reinhardt, Mads Kehlet Jepsen, David Pisinger

Department of Management Engineering, Technical University of Denmark

July 8, 2011

## Abstract

We consider an important generalization of the vehicle routing problem with time windows in which a fixed cost must be paid for accessing a set of edges. This fixed cost could reflect payment for toll roads, investment in new facilities, the need for certifications and other costly investments. The certifications and contributions impose a cost for the company while they also give unlimited usage of a set of roads to all vehicles belonging to the company. Different versions for defining the edge sets are discussed and formulated. A MIP-formulation of the problem is presented, and a solution method based on branch-and-price-and-cut is applied to the problem. The computational results show that instances with up to 50 customers can be solved in reasonable time, and that the branch-cut-and-price algorithm generally outperforms CPLEX. It also seems that instances get more difficult when the penalized edge sets form a spanning tree, compared to when they are randomly scattered.

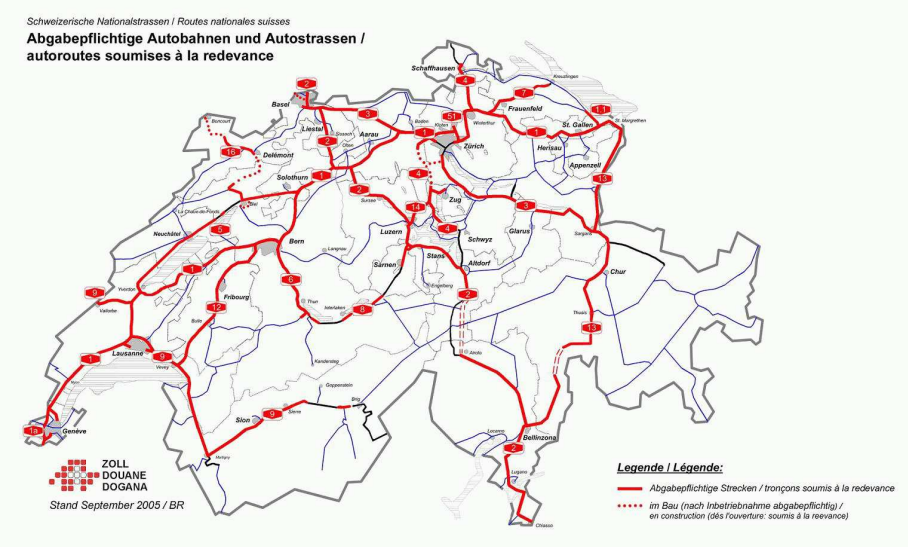



Figure A.1: Main road net in Switzerland. To access all the red edges, a vignette needs to be paid. A transportation company may choose to avoid the toll roads and only use the ordinary highways

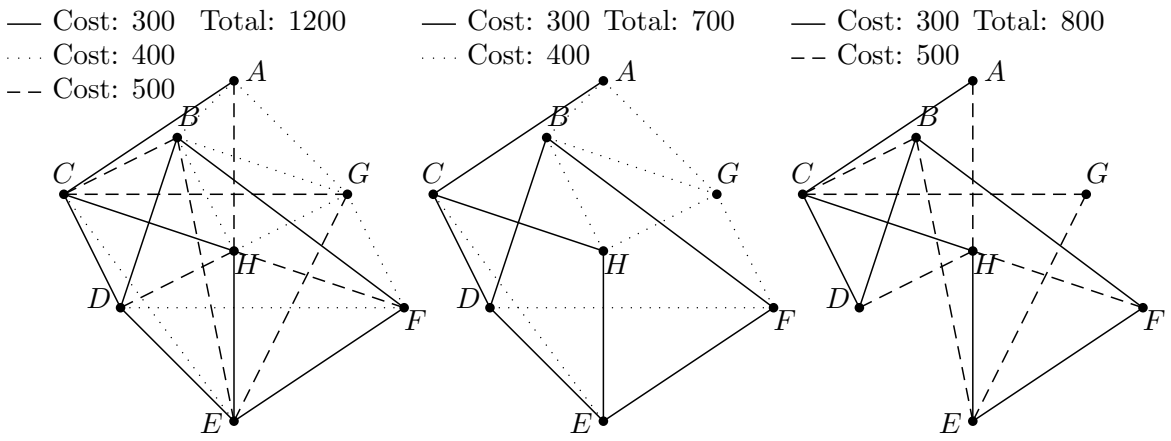
## A.1 Introduction

In certain real-life situations the cost of a connection does not entirely depend on the cost of the individual links (edges). Frequently, in real life, a fee must be paid by the company for allowing its vehicle to access roads, areas, bridges or other. Such a fee may in some cases only be required to be paid once by the company and is in such cases independent of the number of vehicles accessing any edge in the set. Companies routing in an area with many ferry connections may pay to access a set of ferries owned by a company at a monthly rate or at a reduced price. Here, it is important to determine which ferry companies it is most profitable to use. The same applies to Toll roads and bridges, where some countries charge a company based tax for accessing all freeways in the country (see Figure A.1). In war zones or areas of unrest a company may need to get a certification allowing its vehicles to travel on certain protected roads or to enter certain protected zones. Even though in some cases the access is to be paid only for the vehicle accessing the edge set the company will often wish to sign up all its vehicles for robustness and easy administration purposes. Yet another situation where a set of edges can be accessed at a fixed cost is in cases where there is an option of investing in a facility. In such problems, referred to as location-routing problems, there is often a fixed charge connected to a facility and location. Nagy and Salhi [19] give an extensive survey of location-routing problems covering many different routing problems combined with facility location problems. Belenguer et al. [2] recently presented a branch-and-cut method for the location routing problem. In all of the mentioned cases there is a fixed charge for accessing a set of edges. Apart from considering multiple depots, the problem in [2] can be seen as a special case of the model presented in this paper.

The problem of minimizing the overall cost when planning routes that have a cost associated with sets of edges is in this paper investigated as a generalization of the well known problem of routing vehicles with capacity and service time window restrictions (VRPTW).

In the version of the VRPTW considered here the edges of the graph belong to different sets. Once the cost of accessing the set is paid all vehicles can access the edges in the set. However, there might still be a price associated with each of the edges used. Note that the price for accessing the set is paid at most once. This cost has an influence on all the routes since once the access price is paid the edges can be accessed by another vehicle without paying the access price again. This makes the cost of the different vehicle routes interdependent. We will denote the considered problem an *edge set vehicle routing problem with time windows* (ESVRPTW). In Figure A.2 an example of a network with the edges partitioned into different sets is shown. Figure A.2 a) shows the entire set of edges, and b) and c) show accessible edges when paying for different combinations of two edge set. Clearly the ESVRPTW is NP-hard as it is a generalization of the VRPTW problem. We will in this paper present a model for the problem and a Danzig-Wolfe decomposition similar to the classical decomposition of the VRPTW.

The paper is organized as follows. In the following section we give a rough overview of literature for the vehicle routing problem with time windows and describe relevant results that can be used for solving the ESVRPTW. Section A.3 presents a MIP model for the ESVRPTW and in Section A.4 the decomposition of the problem into a Master and subproblem is described. Moreover the solution method and valid inequalities are described. In Section A.5 various extensions of the ESVRPTW model are discussed. In Section A.6 the test instances are described. Section A.7 reports computational results, and finally the paper is concluded




(a) All 3 edge sets of the graph. (b) Accessible edges when the two edge sets shown are paid for. (c) Accessible edges when the two edge sets shown are paid for.

Figure A.2: The graph of a) all three edge sets b) two edge sets and c) another two edge sets

in Section A.8.

## A.2 Literature review

To the best of our knowledge, the problem of routing vehicles with an edge set cost has not yet been investigated in the published literature. However, the underlying problem, the vehicle routing problem with time windows (VRPTW), has been extensively studied. The vehicle routing problem was introduced in 1959 by Danzig and Ramser in [6] as the truck dispatching problem. Many different exact and heuristic methods have been applied to the problem. The basis of the research in this paper is in the exact methods. In 1981 Christofides et al. [4] presented a decomposition generating $q$-routes for the capacitated VRP. One of the first exact methods for the VRPTW was by Kolen et al. [15] using the ideas presented in [4] and applying them to the VRPTW. This was later included in a branch-and-price method by Desrochers et al [9].

In 1987 a benchmark suite was presented for the VRPTW [21] making it easy to compare solution methods and the research society has been enticed by the problem of solving these tests. Recently there has been a strong development in solution times and problem sizes solved to optimality. In 1999 Kohl et al. [14] and Cook and Rich [5] both applied branch-cut-and-price to the VRPTW.

Some of the most recent developments in solving the VRPTW are described in [1], [8],[11], and [13]. Both Jepsen et al. [11] and Baldacci et al. [1] use the valid cuts suggested by Lysgaard et al. [18] to separate candidate sets for branching. Even though the cuts in [18] are implemented for the capacitated vehicle routing problem (CVRP) they may be used for the VRPTW, as the solutions to the VRPTW problem are a subset of the solution to the corresponding CVRP. Jepsen et al. [11] implemented a branch cut and price algorithm with a label-setting bi-directional algorithm for elementary shortest paths developed by Righini and Salani [20]. Jepsen et al. added the subset-row (SR) inequalities on the master problem variables and modified the subproblem to include the reduced cost from these inequalities. These SR inequalities are included by both Desaulniers et al. [8] and Baldacci et al. [1] in their algorithms.

Desaulniers et al. in 2008 [8] further improved the results by using Tabu search for

finding improving routes in the subproblem and generalized the $k$-path inequalities originally formulated by Kohl et al. [14].

Baldacci et al. [1] introduce an enumuration framework. The master problem is solved using a subgradient optimization algorithm and enumaration is done by solving an ESPPRC where standard dominance is limited. To improve the dominance a lower bound for the completion of each label is found using the ng-routes.

In this paper we formulate the ESVRPTW and solve it using the solution method used by Jepsen et al. [11] on the VRPTW as this method can be easily adapted to solve the ESVRPTW.

## A.3 The Model

The mathematical model is based on the model presented in [11]. Given the following sets:

| | |
|---|---|
| $C$ | The set of customers |
| $R$ | The set of edge groups |
| $V$ | The set vertices representing the customers in $C$ and the depot defined as 0 |
| $\mathbf{A}$ | The set of arcs $(i, j)$ in $V$ and $\mathbf{A}_r$ is the set of arcs $(i, j)$ belonging to the group $r \in R$ |
| $K$ | The set of vehicles |

The variables are defined as:

| | |
|---|---|
| $x_{ij}^v$ | Indicator variable indicating if the arc $(i, j)$ is used by vehicle $v \in K$ |
| $y_r$ | Indicator variable which is one if an edge from group $r \in R$ is used and zero otherwise |
| $t_i^v$ | The time vehicle $v$ visits $i \in V$. |

The parameters are defined as:

| | |
|---|---|
| $D$ | The capacity of the vehicles |
| $d_i$ | The demand which must be delivered to vertex $i \in V$. The demand at the depot is zero |
| $a_i$ | The availability time for customer $i \in C$ |
| $b_i$ | The required completion time for customer $i \in C$ |
| $c_{ij}$ | The cost of using an arc $(i, j) \in A$ |
| $c_r$ | The cost of accessing the arcs in group $r \in R$ |

Since the problem is a generalization of the VRPTW the model presented here for the ESVRPTW is the standard VRPTW model presented by Kallehauge in the survey [12], with an additional set of constraints used to formulate the edge set costs. The cost of the edge sets are inserted into the objective. In the presented model the assumption is that each edge belongs to exactly one set, however, alternatives to this assumption are discussed in Section A.5.

$$\textbf{Min:} \sum_{v \in K} \sum_{(i,j) \in \mathbf{A}} c_{ij} x_{ij}^v + \sum_{r \in R} c_r y_r \tag{A.1}$$

$$s.t. \quad y_r - \sum_{v \in K} x_{ij}^v \geq 0 \qquad \forall r \in R,\ (i,j) \in \mathbf{A}_r \qquad \text{(A.2)}$$

$$\sum_{v \in K} \sum_{(i,j) \in \mathbf{A}} x_{ij}^v = 1 \qquad \forall i \in C \qquad \text{(A.3)}$$

$$\sum_{i, \in C} x_{i0}^v = \sum_{i \in C} x_{0i}^v \qquad \forall v \in K \qquad \text{(A.4)}$$

$$\sum_{(ji) \in \mathbf{A}} x_{ji}^v - \sum_{(ij) \in \mathbf{A}} x_{ij}^v = 0 \qquad \forall i \in C, \forall v \in K \qquad \text{(A.5)}$$

$$\sum_{(ij) \in \mathbf{A}} d_i x_{ij}^v \leq D \qquad \forall v \in K \qquad \text{(A.6)}$$

$$a_i \leq t_i^v \leq b_i \qquad \forall\, i \in V, v \in K \qquad \text{(A.7)}$$

$$(t_i^v + \theta_{ij}) x_{ij}^v - t_j^v \leq 0 \qquad \forall\ v \in K,\ (i,j) \in \mathbf{A} \qquad \text{(A.8)}$$

$$x_{ij}^v \in \{0,1\} \qquad \forall\, (i,j) \in \mathbf{A},\ v \in K \qquad \text{(A.9)}$$

$$y_r \in \{0,1\} \qquad \forall\ r \in R \qquad \text{(A.10)}$$

$$t_i^v \in \mathbb{Z}_0^+ \qquad \forall\, i \in V,\ v \in K \qquad \text{(A.11)}$$

The objective (A.1) is the sum of the cost on the edges accessed and the sum of the cost of accessing the sets of the edges accessed. The constraints (A.2) ensure that if an edge in a set is used then the cost of accessing the set is paid. Note that the integrality of the $x_{ij}^v$ variables implies integral $y_r$ variables. Constraints (A.3) ensure that every customer is visited. Constraints (A.4) ensure that all vehicles start and end their journey at the depot. Constraints (A.5) ensure that vehicles arriving at a customer also leaves the same customer. Constraints (A.6) ensure that the capacity of a vehicle is not exceeded. Constraints (A.7) ensure that customers are visited in their respective time window. Finally, constraints (A.8) ensure that the vehicles travel a connected path. The variables $x_{ij}^v$ and $y_r$ are in (A.9) and (A.10) defined to be binary and the variable $t_i^v$ is in (A.11) defined to be a positive integer.

### A.3.1 Tightening of the edge set constraints

In the ESVRPTW each costumer must be visited exactly once. This requirement is ensured by constraints (A.3) and can be used to tighten the constraints in (A.2). Since each costumer is visited once, we know that if several edges belonging to the same set leave the same costumer then at most one of them can be used, and if one of them is used then the cost of the set must be accounted for.

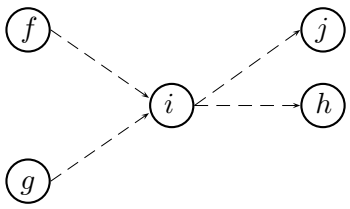



Figure A.3: Bound on outgoing edges

From this observation we can construct the constraints:

$$\sum_{v \in K} \sum_{(i,j) \in \mathbf{A_r}} x_{ij}^v \leq y_r \qquad \forall i \in C, \forall r \in R \tag{A.12}$$

In this case the integrality of the $x$ variables again imposes the integrality of the $y$ variables and the number of constraints in (A.12) is $|C|$. Note that constraints (A.12) do not apply to the depot as more than one edge belonging to a group may leave the depot. Therefore the constraints of type (A.12) cannot entirely replace the constraints (A.2). However, by formulating a new set of constraints (see (A.13)) similar to the constraints (A.12) for edges entering every costumer then the constraints (A.2) can be replaced by $2|C|$ constraints.

This means that the constraints (A.2) in the model can be replaced by the constraints:

$$\sum_{v \in K} \sum_{(j,i) \in \mathbf{A_r}} x_{ji}^v \leq y_r \qquad \forall i \in C, \forall r \in R \tag{A.13}$$

$$\sum_{v \in K} \sum_{(i,j) \in \mathbf{A_r}} x_{ij}^v \leq y_r \qquad \forall i \in C, \forall r \in R \tag{A.14}$$

These tighter constraints will in the following replace constraints (A.2) in the model.

## A.4 Solution method

The branch-cut-and-price method has with success been applied to the VRPTW problem and the best results for finding exact solutions to the problem have been produced using this method (see Jepsen et al. [11], Desaulniers et al. [8] and Baldacci et al. [1]). Since the problem ESVRPTW is a generalization of the VRPTW the solution methods for the VRPTW may successfully be applied to the ESVRPTW. Therefore we will apply the BCP algorithm to the VRPTW using cuts for the original formulation of the VRPTW presented by Fukasawa et al. in [10] and by Lysgaard et al. [18] for the CVRP. This corresponds to the algorithm developed by Jepsen et al. [11] for the VRPTW problem. Jepsen et al. also introduced the Subset Row valid inequalities into the master problem formulation. We will later argue that these cuts can with the same benefits be applied to the ESVRPTW.

The ESVRPTW can be decomposed into a master and pricing problem using the standard VRPTW Dantzig-Wolfe decomposition where, the pricing problem is to find a elementary shortest path problem with resource constraints.

### A.4.1 Master Problem

The master problem is similar to the standard VRPTW decomposition master problem presented by Desrochers et al. [9]. However, the cost of the edge sets are kept in the master problem and these costs will be reflected in the dual variables from the solution of the linearly relaxed master problem.

$$\textbf{Min:} \sum_{p \in P} \sum_{(i,j) \in \mathbf{A}} c_{ij} \alpha_{ijp} \lambda_p + \sum_{r \in R} c_r y_r \tag{A.15}$$

$$s.t. \quad \sum_{p\in P}\sum_{(j,i)\in \mathbf{A_r}} \alpha_{jip}\lambda_p \leq y_r \qquad \forall i \in C, \forall r \in R \qquad \text{(A.16)}$$

$$\sum_{p\in P}\sum_{(i,j)\in \mathbf{A_r}} \alpha_{ijp}\lambda_p \leq y_r \qquad \forall i \in C, \forall r \in R \qquad \text{(A.17)}$$

$$\sum_{p\in P}\sum_{(j,i)\in \mathbf{A}} \alpha_{jip}\lambda_p = 1 \qquad \forall i \in C \qquad \text{(A.18)}$$

$$\lambda_p \in \{0,1\} \qquad \forall\, p \in \mathbf{P} \qquad \text{(A.19)}$$

$$y_r \in \{0,1\} \qquad \forall\ r \in R \qquad \text{(A.20)}$$

The set $P$ contains routes satisfying the time window constraints and the capacity constraints. When $\lambda_p$ is one then route $p \in P$ is used and $\lambda_p$ is zero otherwise. The constant $\alpha_{ijp}$ is one if the edge $(i,j) \in A$ is used by the route $p$ and zero otherwise. Constraints (A.16) and (A.17) corresponds to the constraints (A.13) and (A.14) which ensure that access to the edges used is paid once if an edge is used in one of the selected routes. Constraints (A.18) ensure that every customer is visited exactly once by the set of routes selected. The master problem can be recognized as a set partitioning problem with side constraints. It is important to note that the constraints (A.16) and (A.17) do not change the domain of valid solutions but only affect the value of the solutions.

### A.4.2 Sub problem

The linear relaxation of the master problem can be solved through delayed column generation. The pricing problem is the elementary shortest path problem with resource constraints. Let $\phi'_{ir} \in \mathbb{R}$ be the dual variables of constraints (A.16) and let $\phi_{ir} \in \mathbb{R}$ be the dual variables of constraints (A.17). Let $\pi_j \in \mathbb{R}$ be the dual variables of constraint (A.18) and let $\pi_0 = 0$, $\phi'_{0r} = 0$ and $\phi_{0r} = 0$. Then, the reduced cost for a route in the pricing problem becomes:

$$\bar{c}_p = \sum_{(i,j)\in\mathbf{A}} c_{ij}\alpha_{ijp} - \sum_{(i,j)\in\mathbf{A}} \pi_j\alpha_{ijp} - \sum_{r\in R}\sum_{(j,i)\in A_r} \phi'_{ir}\alpha_{jip} - \sum_{r\in R}\sum_{(i,j)\in A_r} \phi_{ir}\alpha_{ijp} \qquad \text{(A.21)}$$

$$= \sum_{(i,j)\in\mathbf{A}} (c_{ij} - \pi_j)\alpha_{ijp} - \sum_{r\in R}\sum_{(i,j)\in A_r} (\phi_{ir} + \phi'_{jr})\alpha_{ijp} \qquad \text{(A.22)}$$

This can be transformed to the elementary shortest path problem with resource constraints (ESPPRC) where each edge $(i,j)$ has the cost $\bar{c}_{ij} = c_{ij} - \pi_j - \sum_{\{r|(i,j)\in A_r\}}(\phi_{ir} + \phi'_{jr})$. The resource constraints included in the elementary shortest path problem are the demand picked up along the route and the time accumulated along the route. The demand of the customers visited by the route must be less than the capacity and the customers must be visited within their time window. The ESPPRC pricing problem can be solved by a label-setting bidirectional shortest path algorithm developed by Righini and Salani [20]. The domination criteria presented by Desaulniers et al. [7] for removing all labels which are not efficient Pareto optimal, given that the resources are additive or the function on them is strictly de-/in-creasing, can be used here. However, when introducing the SR cuts which will be described later the objective is no longer additive and the function used is not strictly de-/in-creasing. In [3] Blander Reinhardt and Pisinger cover several different ESPPRC problems with objectives containing functions which are not strictly de-/in-creasing.

### A.4.3 Cuts

After adding route variables to the master problem it is investigated if cuts can be added to the master problem. If the added cuts are valid inequalities derived from the original formulation (A.2) to (A.10) then the dual can be transferred directly to the cost of the arcs. Such cuts could be capacity inequalities, strengthened capacity inequalities, framed capacity inequalities, strengthened comb inequalities, multi star inequalities and generalized large multi star inequalities. However, if the cuts added are in the form of the paths variables the dual cost can be more complicating to transfer as the dual of the constraints may be activated not only by a single edge but a combination of edges. However, the subset row cuts have with success been introduced into the master problem variables by Jepsen et al. [11]. Jepsen et al. [11] developed a method of handling the reduced cost of a route for the ESPPRC where the objective contains a function which is not strictly in- or de- creasing as a result of the reduced cost achieved from the Subset Row cuts.

#### Valid Inequalities in the original form

Many valid inequalities have been developed for the VRPTW problem. These valid inequalities will also be applicable for ESVRPTW problem as the ESVRPTW problem does not change the set of feasible solutions but only changes the objective function. Valid inequalities for the VRPTW are described in [10], [16] and [18]. The valid inequalities in the original form applied are, as mentioned previously, the capacity inequality, the strengthened capacity inequality, the framed capacity inequality, the strengthened comb inequality, the multi star inequality and the generalized large multi star inequality. These have all been developed for the capacitated vehicle routing problem CVRP but also apply to the VRPTW. However, since they are developed for the CVRP problem they do not include the time window restrictions to possibly tighten inequalities. The separation algorithm used is that described by Lysgaard et al. [18] and accessible in the framework developed by Lysgaard [17].

#### Valid Inequalities in the master problem form

In [11], Jepsen et al. developed the Subset-Row (SR) inequalities to generate cuts in the set partitioning formulation of the master problem. The SR inequalities are inspired by the clique and odd hole inequalities for the set-packing problem.

The inequalities are not based on the edges directly but on the route variables and formulated as follows:

$$\text{(Subset-Row:)} \quad \sum_{p \in P} \left\lfloor \frac{1}{k} \sum_{i \in S} \alpha_{ip} \right\rfloor \lambda_p \leq \left\lfloor \frac{|S|}{k} \right\rfloor \tag{A.23}$$

Where $S$ is a subset of the constraints (A.18) and $0 < k \leq |S|$ and

$$\left\lfloor \frac{1}{k} \sum_{i \in S} \alpha_{ip} \right\rfloor = \left\lfloor \frac{1}{k} \sum_{i \in S} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp} \right\rfloor$$

Clearly, if $\lambda_p$ has a binary value satisfying the customer constraint (A.18) then the SR inequalities (A.23) will also be satisfied. However, when solving the linearly relaxed master problem $\lambda_p$ are relaxed to linear variables between zero and one then there might be solutions

where the inequalities (A.23) are not satisfied. These inequalities are limited to the set of (A.18) constraints and can therefore easily be introduced in the ESVRPTW problem. Moreover the effect from introducing the SR cuts into the ESVRPTW should be the same as in the VRPTW as the set of feasible solutions do not differ between the ESVRPTW and the VRPTW.

The problem with these inequalities is that the dual of each inequality can not be mapped directly to the cost of the individual edges. Using the notation from Jepsen et al. [11] we let the dual variable of a SR inequality be $\sigma$ we can then formulate the dual cost of a route $p$ as $\hat{c}_p = \bar{c}_p + \sigma \left\lfloor \left( \sum_{i \in S} \sum_{(i,j) \in \delta^+(i)} \alpha_{ijp} \right) / k \right\rfloor$. Note that the reduced cost $\sigma$ is not activated before at least $k$ vertices in the set $S$ has been visited. Therefore to introduce the reduced cost of these constraints into the pricing problem the ESPPRC needs to be modified. As mentioned in Section A.4.2, the ESPPRC is solved with a label-setting algorithm using domination for the time, load and cost criteria. Let $L$ be a label at a node $v$ so that each label $L$ at $v$ represents a path from the depot to $v$. The usual dominance criteria which holds for additive costs is that a label is dominated if there exists another label at the same vertex where all criteria are less than or equal to the dominated labels criteria values. However, this does not hold for the reduced cost introduced by the SR inequalities. One label may be better than the other even if the labels have the same or worse cost. We work with the cuts not allowing two or more routes to visit two vertices from a set of three customers, $k = 2$ and $|S| = 3$.

To solve this problem Jepsen et al. [11] modified the domination rule for the cost criteria in the ESPPRC. The modification consists of subtracting the reduced cost $\sigma_q$ from a label $L_i$. Where the cost of cut $q$ is included in label $i$ and not included in dominated label $L_j$ so that the domination rule for the cost of two labels at the same vertex becomes:

$$\hat{c}(L_i) - \sum_{q \in Q} \sigma_q \leq \hat{c}(L_j)$$

Where $Q$ is the set of SR cuts with sets of three customers where $L_i$ has visited two vertices in the set $S$ and $L_j$ has not and $\sigma_q < 0$ . The domination rules for the remaining constraints are kept the same. For further details see Jepsen et al. [11].

### A.4.4 Branch-and-cut-and-price

The branch-cut-and-price algorithm is commonly used for solving integer problems to optimality. Below we describe the algorithm with some of the conditions selected in the method used here included.

The branch-cut-and-price algorithm:

Step 1: Choose an unprocessed node. The node with the lowest lower bound. If the lower bound of a node is above the upper bound then the node will be removed from the unprocessed node list.

Step 2: Solve the LP relaxed master problem.

Step 3: search for routes with negative reduced cost using heuristic methods. If any found add up to 400 to the master problem and go to step 2. The heuristic used is a simplified version of the ESPPRC algorithm.

Step 4: Solve the pricing problem to optimality. If routes with negative reduced cost are found then add them to the master problem and go to step 2. If no routes with negative reduced cost are found then update the lower bound if the lower bound above the upper bound then remove the node from the unprocessed node list and goto step 1.

Step 5: If any violated cuts are found then add them to the master problem and go to step 2

Step 6: Mark the node as processed. If the solution to the LP relaxed master problem is integer then update the upper bound. If the solution is fractional then branch and add the children to the list of unprocessed nodes. Go to step 1.

The branching used is described in more detail in the next subsection.

### A.4.5 Branching

For VRPTW the branching is most commonly done on edge variables. We have chosen to do branching on the group variables as well. Branching on the group variables can reduce the depth of the search tree as the edges to branch on are restricted. Moreover the number of group variables is comparably small. The node with the lowest lower bound is chosen for branching. The branching candidates are selected as the groups first and otherwise the edges where they are separated by using the branching strategy presented by Fukasawa et al. [10] where branching occurs on a set of customers $S \subset C$ by having one branch with one vehicle covering the set $S$. This is represented by constraint $\sum_{v \in K} \sum_{(i,j) \in \delta^+(S)} (x_{ij}^v + x_{ji}^v) = 2$ where $\delta^+(S)$ is the edges leaving the set $S$. The other branch has at least two vehicles covering the customers in the set $S$ represented by constraint $\sum_{v \in K} \sum_{(i,j) \in \delta^+(S)} (x_{ij}^v + x_{ji}^v) \geq 4$. To separate candidate sets the Lysgaard cut library [17] is used. From preliminary tests it was clear that branching on the group variables tended to improve the solution time for the problem significantly.

## A.5 Closely related formulation and problems

Clearly, there are different versions of the problem where the edges are part of a set. It has been assumed that the edges only belong to one set, however there could be cases where the edges belong to more than one set. If an edge can belong to one set there can be different ways to add the charge. The simple choice would be that the cost must be paid for all the sets an edge belongs to. This problem can be formulated with the constraints (A.13) and (A.14) with the only difference that an edge may be included in more than one constraint for each node. Other alternatives are discussed in this section.

### A.5.1 Edges belonging to multiple sets

Another variant could be that for an edge belonging to several sets, the access cost only needs to be paid for one of the sets to which the edge belongs. This can be formulated as:

$$x_{ij} - \sum_{(i,j) \in r \in R} y_r \leq 0 \ \ \forall (i,j) \in \mathbf{A} \tag{A.24}$$

This constraint is very similar to constraints (A.2); however, in this case the integrality of the $x_{ij}$ variables does not necessarily imply integrality of the $y_r$ variables. Note, that when replacing constraints (A.16) and (A.17) with (A.24) each edge $(i, j)$ in the ESPPRC sub problem will have cost $c_{ij} - \pi_j - \theta_{ij}$ where $\theta$ is the dual variable for the constraints (A.24). This will not add any complications to the ESPPRC algorithm as the cost of a path remains additive and the non additive cost introduced by the SR cuts are handled as in the VRPTW.

### A.5.2 Accessing a set of reduced prices

In some cases one may access edges belonging to a set without paying for accessing the set but by paying a more expensive price for using each edge. For instance, instead of buying company access to all freeways in a country, one may be allowed to pay with cash at the barrier to each road. These cash prices are expensive but may be attractive if there is a very limited usage of the edges in the set. This extension is easily handled in our model by duplicating each edge, where one edge belongs to an edge set, and the other edge correspond to cash payment.

$$x_{ij}^{z_r} - \sum_{r \in (i,j)} y_r \leq 0 \;\; \forall (i,j) \in \mathbf{A} \tag{A.25}$$

where $x_{ij}^{z_r}$ is the edge between $i$ and $j$ which becomes accessible when paying the price for accessing the set $r$.

In this case each edge $x_{ij}^{z_r}$ in the ESPPRC will have the cost $c_{ij}^{z_r} - \pi_j - \zeta_{ij}$ where $\zeta_{ij}$ is the dual variable of the respective constraint of type (A.25) and the cost of the edges not in sets will simply have the cost $c_{ij} - \pi_j$.

## A.6 Test data

Following the tradition in VRP problems, the test data are based on the Solomon instances [21] making it possible to relate our results to the existing literature. We have generated test instances based on the RC201 to RC204 and C101 to C109 instances by assigning subsets of edges to disjoint groups, and associating a fixed cost with each group. For the RC201 to RC204 Solomon instances different categories of test instances have been constructed. The instances can be grouped into two categories:

1. **random sets**: In these instances, the edges in each set are randomly selected. These instances should reflect a toll on accessing bridges, tunnels or ferries. These facilities are randomly scattered in the plane, but frequently a set of facilities is run by the same operator.

2. **spanning tree sets**: In these instances the selected edges form cheap spanning graphs of a randomly selected subset of vertices. Each subset of vertices consists of half of the total number of vertices. This case should reflect payment of toll on motorways. Motorways usually form a spanning network covering the main cities in a country.

In all test cases, each edge is assigned to at most one edge set.

For each Solomon instance, test instances containing 3, 5 and 8 edge sets were generated, each having an associated cost for accessing the set.

For the **random edge sets** instances, 50% of the edges are assigned to groups with an additional cost. For each combination of Solomon instance and number of groups, two test instances were generated: one case with the costs of an edge set group calculated as $\beta = 5\%$ of the average cost of the edges in the group multiplied by the number of vertices in the set, and another case using the same calculations with $\beta = 10\%$.

In the case of **spanning tree sets**, the cost of a given set is chosen as the most expensive edge in the graph minus the average value of the edges in the given set. This implies that sets containing cheaper spanning trees (i.e. fast transportation times) are more costly than the sets containing more expensive spanning trees.

For the Solomon instances RC201 to RC204, test cases were generated with 15, 20, 30 and 40 customers. For the instances C101 to C109, test cases were generated containing 50 customers using **random edge sets**. We only consider test cases up to 40 customers for the RC201 to RC204 instances, since many instances with 40 customers were not solveable within the given time limit. Instead we have run larger instances with 50 customers for the C101 to C109 instances, as these are know to be easier from the VRPTW literature.

## A.7 Results

The program has been implemented in C++ using the COIN bcp library and CLP as the linear programming solver. The test have been run on a Linux machine with a 64 bit Intel Xeon 2.67 GHz CPU. The edge set constraints have been implemented in the framework by Jepsen et al. [11] provided to us by the authors. On the RC202 - RC204 tests with 20 customers we have tested the effect of running branch-cut-and-price with Lysgaard [17] cuts only, the SR cuts [11] only, both the Lysgaard and SR cuts, and without any cuts. In Table A.1 the solution times for the four algorithms and for CPLEX are shown. In about half of the instances, CPLEX is not able to solve the routing problem within the time limit. For all four branch-and-price and branch-cut-and-price algorithms the solution was found within 500 seconds. We have ranked the results of the four branch-cut-and-price algorithms by solution times. In Table A.1 the rank of a solution is stated in paranthesis after the solution time. The average of the ranks is calculated for each solution algorithm and shown in the last line of Table A.1.

It is seen that CPLEX is the fastest for 7 instances, while the branch-cut-and-price algorithm using both Lysgaard cuts and SR cuts is the fastest for 11 instances. The branch-cut-and-price algorithms using only one of the cut families are only fastest for 3 instances. The ranking average clearly shows that the branch-cut-and-price algorithm using both Lysgaard and SR cuts has the best average rank. Therefore the branch-cut-and-price algorithm used for the tests in Tables A.2, A.3, A.4 and A.5 includes the Lysgaard and SR cuts.

Tables A.2, A.3 and A.4 consider test instances with **random sets**. In Table A.2 and Table A.3 it is seen that the branch-cut-and-price with both Lysgaard and SR cuts often has a significantly reduced running time. However, for all of the instances based on RC201 CPLEX finds the solutions within seconds and always much faster than the branch-cut-and-price algorithm. In Table A.2 there are two instances with 30 customers which cannot be solved within 7500 seconds using the branch-cut-and-price algorithm. However, more than half of the instances cannot be solved by CPLEX within the time limit of 7500 seconds. For the RC201 to RC204 instances with 40 customers shown in Table A.3 only the RC201 instances were solved by the branch and bound algorithm within the time limit, and 9 instances were

| instance | | | opt solution | | solution times | | | | |
|---|---|---|---|---|---|---|---|---|---|
| test | groups | cost $\beta$ | objective | groups | CPLEX | bcp L+SR | Bp | bcp SR | bcp L |
| rc201 | 3 | 05 | 4239 | 3 | *0.06 | 1.22 (3) | 1.16(1) | 1.22(3) | 1.20(2) |
| rc201 | 3 | 10 | 4912 | 2 | *0.06 | 0.97(1) | 0.99(2) | 1.01(3) | 1.05(4) |
| rc201 | 5 | 05 | 4539 | 4 | *0.06 | 1.09(1) | 1.23(4) | 1.18(3) | 1.16(2) |
| rc201 | 5 | 10 | 5787 | 4 | *0.1 | 10.43(3) | 7.36(1) | 10.83(4) | 7.86(2) |
| rc201 | 8 | 05 | 4878 | 5 | *0.09 | 1.62(2) | 1.52(1) | 1.66(3) | 1.71(4) |
| rc201 | 8 | 10 | 5847 | 3 | *0.05 | 4.880(2) | 4.030(1) | 5.04(4) | 4.14(2) |
| rc202 | 3 | 05 | 3723 | 2 | 473.82 | *13.55(1) | 14.86(2) | 16.85(3) | 21.76(4) |
| rc202 | 3 | 10 | 4371 | 2 | 200.65 | 11.22(3) | *10.17(1) | 14.49(4) | 10.85(2) |
| rc202 | 5 | 05 | 4120 | 3 | 200.19 | 10.26(2) | *9.40(1) | 11.71(4) | 11.51(3) |
| rc202 | 5 | 10 | 4969 | 2 | 96.86 | 32.47(3) | 13.83(2) | 56.17(4) | *13.44(1) |
| rc202 | 8 | 05 | 4166 | 3 | 25.54 | *9.96(1) | 10.13(2) | 17.06(4) | 11.75(3) |
| rc202 | 8 | 10 | 5115 | 4 | *8.07 | 37.30(3) | 24.54(1) | 45.20(4) | 25.13(2) |
| rc203 | 3 | 05 | 3371 | 1 | - | *29.07(1) | 29.19(2) | 30.30(3) | 54.01(4) |
| rc203 | 3 | 10 | 3694 | 1 | - | *25.08(1) | 28.43(3) | 29.38(4) | 27.85(2) |
| rc203 | 5 | 05 | 3635 | 2 | - | *44.34(1) | 44.66(2) | 47.71(3) | 58.24(4) |
| rc203 | 5 | 10 | 3968 | 1 | 3571.92 | *24.33(1) | 27.76(3) | 32.11(4) | 26.28(2) |
| rc203 | 8 | 05 | 3545 | 2 | - | *75.54(1) | 81.25(2) | 88.53(3) | 100.63(4) |
| rc203 | 8 | 10 | 3954 | 1 | 6294.51 | *42.05(1) | 44.43(3) | 48.35(4) | 43.32(2) |
| rc204 | 3 | 05 | 3148 | 1 | - | *130.83(1) | 147.04(3) | 183.59(4) | 136.88(2) |
| rc204 | 3 | 10 | 3471 | 1 | - | 452.81(4) | 112.62(2) | 303.24 (3) | *89.45(1) |
| rc204 | 5 | 05 | 3414 | 2 | - | *123.00(1) | 205.80(3) | 153.72(2) | 308.24(4) |
| rc204 | 5 | 10 | 3797 | 1 | - | 315.28(3) | 121.67(2) | 397.13(4) | *106.00(1) |
| rc204 | 8 | 05 | 3215 | 1 | - | 100.40(3) | *43.06(1) | 119.16(4) | 54.43(2) |
| rc204 | 8 | 10 | 3507 | 1 | - | *46.61(1) | 61.35(4) | 54.26(2) | 60.78(3) |
| Average Rank | | | | | | 1.88 | 2.04 | 3.46 | 2,58 |

Table A.1: RC201-RC204 instances with 20 customers, **random sets**. If the algorithm has not terminated within 7500 seconds it is indicated with "-". The best running time for each instance is marked with a "*". L indicates that the cuts implemented in Lysgaard are used, SR indicates that SR-cuts are used, while SR+L indicates that both families of cuts are used.

| instance | | | opt solution | | solution times | |
|---|---|---|---|---|---|---|
| test | groups | cost $\beta$ | objective | groups | CPLEX | Bcp L+SR |
| rc201 | 3 | 05 | 7100 | 3 | *1.08 | 9.30 |
| rc201 | 3 | 10 | 8396 | 1 | *0.7 | 20.78 |
| rc201 | 5 | 05 | 7173 | 2 | *0.14 | 13.75 |
| rc201 | 5 | 10 | 8512 | 1 | *0.23 | 21.60 |
| rc201 | 8 | 05 | 7685 | 4 | *0.41 | 28.31 |
| rc201 | 8 | 10 | 9031 | 2 | *0.64 | 65.73 |
| rc202 | 3 | 05 | 5660 | 2 | - | *52.90 |
| rc202 | 3 | 10 | 6430 | 1 | 283.68 | *167.84 |
| rc202 | 5 | 05 | 5886 | 2 | - | *206.10 |
| rc202 | 5 | 10 | 6831 | 1 | 919.49 | *253.07 |
| rc202 | 8 | 05 | 5915 | 3 | 2897.36 | *128.22 |
| rc202 | 8 | 10 | 7398 | 2 | *328.52 | 3407.25 |
| rc203 | 3 | 05 | 5144 | 1 | - | *104.29 |
| rc203 | 3 | 10 | 5914 | 1 | - | *119.07 |
| rc203 | 5 | 05 | 5429 | 1 | - | *509.70 |
| rc203 | 5 | 10 | 6215 | 1 | - | *1533.37 |
| rc203 | 8 | 05 | 5821 | 2 | - | *870.11 |
| rc203 | 8 | 10 | 6494 | 0 | - | *1491.57 |
| rc204 | 3 | 05 | 4847 | 1 | - | *262.21 |
| rc204 | 3 | 10 | 5587 | 1 | - | *545.37 |
| rc204 | 5 | 05 | - | - | - | - |
| rc204 | 5 | 10 | 5743 | 1 | - | *1621.71 |
| rc204 | 8 | 05 | - | - | - | - |
| rc204 | 8 | 10 | 6118 | 0 | - | *3307.06 |

Table A.2: RC201-RC204 instances with 30 customers, **random sets**. If the algorithm has not terminated within 7500 seconds it is indicated with "-". The best running time is marked with a "*".

| instance | | | opt solution | | solution times | |
|---|---|---|---|---|---|---|
| test | groups | cost $\beta$ | objective | groups | CPLEX | Bcp L+SR |
| rc201 | 3 | 05 | 8660 | 2 | *0.47 | 7.73 |
| rc201 | 3 | 10 | 10730 | 2 | *0.95 | 28.95 |
| rc201 | 5 | 05 | 9431 | 3 | *1.14 | 62.16 |
| rc201 | 5 | 10 | 11982 | 2 | *6.63 | 127.22 |
| rc201 | 8 | 05 | 10064 | 3 | *4.85 | 152.61 |
| rc201 | 8 | 10 | 11922 | 2 | *2.45 | 348.53 |
| rc202 | 3 | 05 | 7805 | 1 | - | *226.13 |
| rc202 | 3 | 10 | 8841 | 1 | - | *503.73 |
| rc202 | 5 | 05 | 8518 | 2 | - | *737.95 |
| rc202 | 5 | 10 | 10072 | 1 | - | *3671.59 |
| rc202 | 8 | 05 | 8755 | 2 | - | *4316.18 |
| rc202 | 8 | 10 | - | - | - | - |
| rc203 | 3 | 05 | 7098 | 1 | - | *430.43 |
| rc203 | 3 | 10 | - | - | - | - |
| rc203 | 5 | 05 | 7120 | 1 | - | *1647.74 |
| rc203 | 5 | 10 | - | - | - | - |
| rc203 | 8 | 05 | - | - | - | - |
| rc203 | 8 | 10 | - | - | - | - |
| rc204 | 3 | 05 | - | - | - | - |
| rc204 | 3 | 10 | 5838 | 0 | - | *882.34 |
| rc204 | 5 | 05 | - | - | - | - |
| rc204 | 5 | 10 | 5838 | 0 | - | *1848.44 |
| rc204 | 8 | 05 | - | - | - | - |
| rc204 | 8 | 10 | - | - | - | - |

Table A.3: RC201-RC204 instances with 40 customers, **random sets**. If the algorithm has not terminated within 7500 seconds it is indicated with "-". The best running time is marked with a "*".

not solved by the branch-cut-and-price algorithm within the time limit.

Table A.2 and Table A.3 show the running time for instances generated from RC201 to RC204 with respectively 30 customers and 40 customers. The running times in Table A.2 show that the branch-cut-and-price for most of the instances runs much faster than CPLEX. The same is true for the running times in Table A.3 when only considering instances where at least one of the algorithms terminated within the time limit.

Notice, that most of the instances not solved by CPLEX within the time limit were solved by branch-cut-and-price. Half of the instances were solved in less than 600 seconds (10 minutes).

For the C101-C109 instances with 50 customers shown in Table A.4 the results are more mixed. CPLEX is the fastest for easy instances, while the branch-cut-and-price algorithm has some computational overhead which only pays off for the difficult instances. The branch-cut-and-price algorithm solves significantly more instances within the time limit, although there are two instances solved by CPLEX which cannot be solved by branch-cut-and-price within the time limit.

Table A.5 contains the test results for instances RC201 to RC204 with **spanning tree sets**. For the RC201 instances, CPLEX solves the instances within a second and considerably faster than the branch-cut-and-price algorithm. For the RC202 to RC204 instances the branch-cut-and-price algorithm solves the problem faster than CPLEX. The last instance has not been solved within the time limit by any of the algorithms. In general the branch-cut-and-price algorithm solves considerably more problems to optimality than CPLEX within the given time limit.

| Instance | | | opt solution | | Solution Times | |
|---|---|---|---|---|---|---|
| test | groups | cost $\beta$ | objective | groups | CPLEX | Bcp L+SR |
| c101 | 3 | 05 | 5683 | 3 | *0.99 | 37.61 |
| c101 | 3 | 10 | 7052 | 3 | *0.66 | 162.79 |
| c101 | 5 | 05 | 6749 | 2 | *1.04 | 574.34 |
| c101 | 5 | 10 | 7725 | 3 | *1.08 | 135.60 |
| c101 | 8 | 05 | 7125 | 3 | *2.27 | 1037.11 |
| c101 | 8 | 10 | 8138 | 2 | *1.4 | 570.75 |
| c102 | 3 | 05 | 5304 | 2 | - | *322.95 |
| c102 | 3 | 10 | - | - | - | - |
| c102 | 5 | 05 | 5741 | 2 | - | *1210.31 |
| c102 | 5 | 10 | - | - | - | - |
| c102 | 8 | 05 | 6201 | 1 | - | *6932.81 |
| c102 | 8 | 10 | - | - | - | - |
| c103 | 3 | 05 | 4801 | 1 | - | *2158.50 |
| c103 | 3 | 10 | 5081 | 0 | - | *535.76 |
| c103 | 5 | 05 | 4979 | 1 | - | *5999.15 |
| c103 | 5 | 10 | 5081 | 0 | - | *252.92 |
| c103 | 8 | 05 | 5081 | 0 | - | *854.73 |
| c103 | 8 | 10 | 5081 | 0 | - | *923.91 |
| c104 | 3 | 05 | - | - | - | - |
| c104 | 3 | 10 | - | - | - | - |
| c104 | 5 | 05 | - | - | - | - |
| c104 | 5 | 10 | - | - | - | - |
| c104 | 8 | 05 | - | - | - | - |
| c104 | 8 | 10 | - | - | - | - |
| c105 | 3 | 05 | 5421 | 2 | *36.90 | 394.63 |
| c105 | 3 | 10 | 6228 | 0 | *121.33 | - |
| c105 | 5 | 05 | 5838 | 2 | *705.29 | 1686.57 |
| c105 | 5 | 10 | 6228 | 0 | *124.26 | 1932.97 |
| c105 | 8 | 05 | 6083 | 1 | *852.82 | 5477.76 |
| c105 | 8 | 10 | 6228 | 0 | *31.51 | 2906.03 |
| c106 | 3 | 05 | 5600 | 2 | *2.05 | 147.36 |
| c106 | 3 | 10 | 6870 | 1 | *3.05 | 458.43 |
| c106 | 5 | 05 | 6428 | 2 | *5.82 | 2652.120 |
| c106 | 5 | 10 | 7317 | 1 | *2.65 | 6803.880 |
| c106 | 8 | 05 | 6896 | 2 | *4.53 | 1340.30 |
| c106 | 8 | 10 | 7378 | 0 | *1.66 | 1453.82 |
| c107 | 3 | 05 | 5204 | 2 | *364.35 | 439.94 |
| c107 | 3 | 10 | 6032 | 0 | *1506.13 | - |
| c107 | 5 | 05 | 5618 | 1 | *2250.54 | 2945.98 |
| c107 | 5 | 10 | 6032 | 0 | *3385.01 | 7312.24 |
| c107 | 8 | 05 | 5700 | 1 | *220.37 | 1164.84 |
| c107 | 8 | 10 | 6032 | 0 | 5519.13 | *1692.73 |
| c108 | 3 | 05 | 5076 | 1 | - | *1260.62 |
| c108 | 3 | 10 | - | - | - | - |
| c108 | 5 | 05 | 5325 | - | - | *1124.48 |
| c108 | 5 | 10 | 5747 | 0 | - | *3845.29 |
| c108 | 8 | 05 | 5628 | 1 | - | *2362.47 |
| c108 | 8 | 10 | 5747 | 0 | - | *5474.220 |
| c109 | 3 | 05 | - | - | - | - |
| c109 | 3 | 10 | - | - | - | - |
| c109 | 5 | 05 | - | - | - | - |
| c109 | 5 | 10 | 5022 | 0 | - | *428.11 |
| c109 | 8 | 05 | 5022 | 0 | - | *2730.11 |
| c109 | 8 | 10 | 5022 | 0 | - | *804.76 |

Table A.4: C101-C107 instances with 50 customers, **random sets**. If the algorithm has not terminated within 7500 seconds it is indicated with "-". The best running time is marked with a "*".

## A.8 Conclusion

The vehicle routing problem with time windows and fixed costs for accessing an edge set (ESVRPTW) has been presented in this paper. To the best of our knowledge, it is the first

| instance | | | opt solution | | solution times | |
|---|---|---|---|---|---|---|
| test | customers | groups | objective | groups | CPLEX | Bcp SR+L |
| rc201 | 15 | 3 | 2618 | 1 | *0.05 | 0.75 |
| rc201 | 15 | 5 | 3153 | 3 | *0.04 | 1.98 |
| rc201 | 15 | 8 | 3474 | 4 | *0.06 | 1.92 |
| rc202 | 15 | 3 | 2478 | 2 | 176.17 | *5.03 |
| rc202 | 15 | 5 | 2773 | 2 | 33.74 | *9.13 |
| rc202 | 15 | 8 | 3154 | 4 | 93.06 | *40.43 |
| rc203 | 15 | 3 | 2400 | 0 | 556.14 | *5.37 |
| rc203 | 15 | 5 | 2665 | 1 | 268.44 | *16.88 |
| rc203 | 15 | 8 | 2982 | 2 | 4342.91 | *151.65 |
| rc204 | 15 | 3 | 2240 | 0 | - | *6.21 |
| rc204 | 15 | 5 | 2526 | 1 | 6698.49 | *36.80 |
| rc204 | 15 | 8 | 2859 | 2 | 2438.21 | *402.83 |
| rc201 | 20 | 3 | 3733 | 2 | *0.02 | 0.26 |
| rc201 | 20 | 5 | 4302 | 3 | *0.03 | 1.34 |
| rc201 | 20 | 8 | 4931 | 3 | *0.05 | 2.42 |
| rc202 | 20 | 3 | 3464 | 1 | 107.58 | *3.88 |
| rc202 | 20 | 5 | 3862 | 2 | 56.45 | *1.25 |
| rc202 | 20 | 8 | 4635 | 4 | 1276.84 | *163.60 |
| rc203 | 20 | 3 | 3042 | 0 | - | *7.09 |
| rc203 | 20 | 5 | 3366 | 1 | - | *136.39 |
| rc203 | 20 | 8 | 4120 | 3 | - | *2986.15 |
| rc204 | 20 | 3 | 2845 | 0 | - | *6.82 |
| rc204 | 20 | 5 | 3301 | 1 | - | *1160.77 |
| rc204 | 20 | 8 | 3790 | 2 | - | *6469.83 |
| rc201 | 30 | 3 | 5599 | 1 | *0.11 | 4.44 |
| rc201 | 30 | 5 | 6204 | 2 | *0.34 | 12.57 |
| rc201 | 30 | 8 | 6998 | 4 | *0.24 | 37.11 |
| rc202 | 30 | 3 | 4832 | 1 | - | *21.95 |
| rc202 | 30 | 5 | 5478 | 2 | - | *99.57 |
| rc202 | 30 | 8 | 6431 | 5 | - | *1482.03 |
| rc203 | 30 | 3 | 4418 | 1 | - | *14.01 |
| rc203 | 30 | 5 | 5191 | 2 | - | *374.77 |
| rc203 | 30 | 8 | 5877 | 4 | - | *4458.23 |
| rc204 | 30 | 3 | 4292 | 0 | - | *24.82 |
| rc204 | 30 | 5 | 4953 | 2 | - | *5606.51 |
| rc204 | 30 | 8 | - | - | - | - |

Table A.5: RC201-RC204 instances, **spanning tree sets**. If the algorithm has not terminated within 7500 seconds it is indicated with "-". The best running time is marked with a "*".

time this type of problem has been investigated. A mathematical model has been presented for the ESVRPTW. We have applied the branch-cut-and-price method to the problem and shown that including the SR cuts and the cuts implemented in Lysgaard [17] for the VRPTW and CVRP improves the solution times for this problem. Many related routing problems may with advantage be implemented this way using the extensive research available for the CVRP and VRPTW problems.

## Acknowledgments

The authors wish to thank Richard Martin Lusby for valuable comments.

# Bibliography


[1] R. Baldacci, A. Mingozzi, and R. Roberti. Solving the vehicle routing problem with time windows using new state space relaxation and pricing strategies. Submitted.

[2] J.-M. Belenguer, E. Benavent, C. Prins, C. Prdhon, and R. W. Calvo. A branch-and-cut method for the capacitated location-routing problem. *Computers & Operations Reserach*, 38:931–941, 2011.

[3] L. Blander Reinhardt and D. Pisinger. Multi-objective and multi-constraint non-additive shortest path problems. *Computers and Operations Research*, 38:605–616, 2011.

[4] N. Christofides, A. Mingozzi, and P. Toth. State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11:145–164, 1981.

[5] W. Cook and J. L. Rich. A parallel cutting-plane algorithm for the vehicle routing problem with time windows. Technical report, Rice University, 1999.

[6] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6:80–91, 1959.

[7] G. Desaulniers, J. Desrosriers, J. Ioachim, I. M. Solomon, F. Soumis, and D. Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems, 1998.

[8] G. Desaulniers, F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized $k$-path inequalities for the vehicle reouting problem with time windows. *Transportation Science*, 42:387–404, 2008.

[9] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40:342–354, 1992.

[10] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragao, M. Reis, E. Uchoa, and R. F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Math. Programming*, 106:491–511, 2006.

[11] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, 56:497–511, 2008.

[12] B. Kallehauge. Formulations and exact algorithms for the vehicle routing problem with time windows. *Computers & Operations Research*, 35:2307–2330, 2008.

[13] B. Kallehauge, J. Larsen, and O. B. G. Madsen. Lagrangian duality applied to the vehicle routing problem with time windows. *Computers & Operations Research*, 33:1464–1487, 2006.

[14] N. Kohl, J. Desrosiers, O. B. G. Madsen, M. M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33:101–116, 1999.

[15] A. W. J. Kolen, A. H. G. Rinnooy Kan, and H. W. J. M. Trienekens. Vehicle routing with time windows. *Operations Research*, 35:266–273, 1987.

[16] A. N. Letchford, R. W. Eglese, and J. Lysgaard. Multistarts, partial multistars and the capacitated vehicle routing problem. *Math. Programming*, 94:21–40, 2002.

[17] J. Lysgaard. Cvrpsep: A package of separation routines for the capacitated vehicle routing problem, 2003.

[18] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming Serie A*, 100:423–445, 2004.

[19] G. Nagy and S. Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177:649–672, 2007.

[20] G. Righini and M. Salani. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3:255–273, 2006.

[21] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35:254–265, 1987.

# Appendix B A Path Based Model for a Green Liner Shipping Network Design Problem

Mads K. Jepsen, Berit Løfstedt, Christian E. M. Plum , David Pisinger [1] and Mikkel M. Sigurd [23]

July 8, 2011

## Abstract

Liner shipping networks are the backbone of international trade providing low transportation cost, which is a major driver of globalization. These networks are under constant pressure to deliver capacity, cost effectiveness and environmentally conscious transport solutions. This article proposes a new path based **MIP** model for the Liner shipping Network Design Problem minimizing the cost of vessels and their fuel consumption facilitating a green network. The proposed model reduces problem size using a novel aggregation of demands. A decomposition method enabling delayed column generation is presented. The subproblems have similar structure to **Vehicle Routing Problems**, which can be solved using dynamic programming.

# Appendix B A Path Based Model for a Green Liner Shipping Network Design Problem

Mads K. Jepsen, Berit Løfstedt, Christian E. M. Plum , David Pisinger [4] and Mikkel M. Sigurd [56]

July 8, 2011

## B.1 Introduction

Global liner shipping companies provide port to port transport of containers, on a network which represents a billion dollar investment in assets and operational costs.

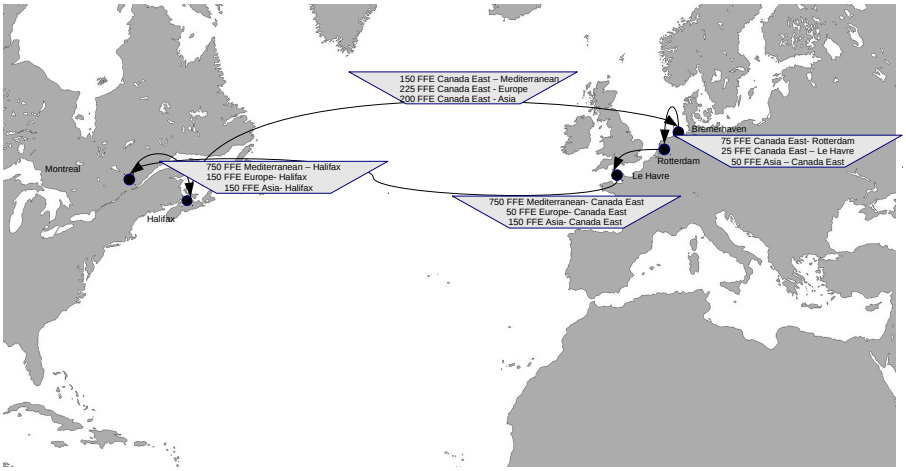



Figure B.1: A Canada-Northern Europe service. FFE is Forty Foot Equivalent unit container used to express the volume of containers in each cargo category.

The liner shipping network can be viewed as a transportation system for general cargo not unlike an urban mass transit system for commuters, where each route (service) provides transportation links between ports and the ports allow for transhipment in between routes (services). The liner shipping industry is distinct from other maritime transportation modes primarily due to a fixed public schedule with weekly frequency of port calls as an industry standard (Stopford [1998]). The network consists of a set of *services*. A *service* connects a sequence of ports in a cycle at a given frequency, usually weekly. In Figure B.1 a service connecting Montreal-Halifax and Europe is illustrated. The weekly frequency means that several vessels are committed to the service as illustrated by Figure B.1, where four vessels cover a round trip of 28 days placed with one week in between vessels. This round trip for the vessel is referred to as a *rotation*. Note that the Montreal service carries cargo to North Europe, the Mediterranean and Asia, with the two latter transhipping in Bremerhaven. Vice versa for cargo headed for Canada has multiple origins. This illustrates that transhipments to other connecting services is at the core of liner shipping. Therefore, the design of a service is complex, as the set of rotations and their interaction through transhipment is a transportation system extending the supply chains of a

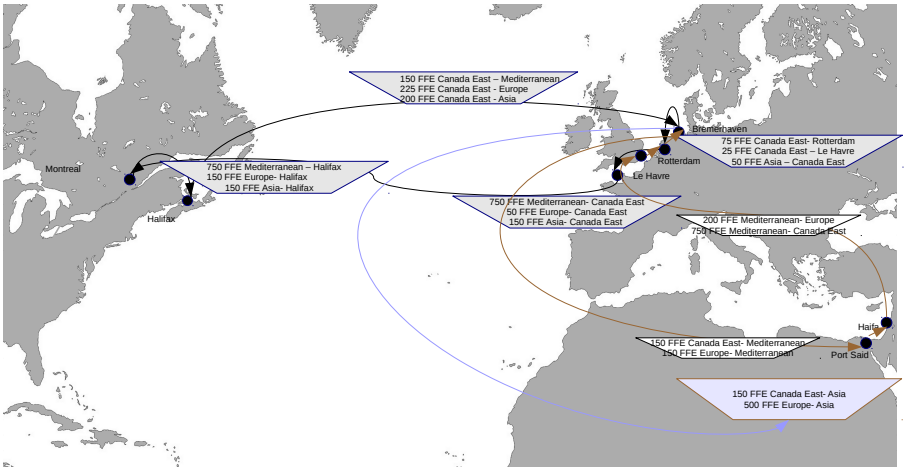



Figure B.2: Two connecting services. The Montreal service from Figure B.1 and a Europe-Mediterranean service with a round trip time of 2 weeks illustrated by two white vessels. The cargo composition on board vessels illustrate transhipments at the core of the liner shipping network design. The light blue incomplete service illustrates a larger service transporting cargo between Europe and Asia.

multiplum of businesses. Figure B.2 illustrates two services interacting in transporting goods between Montreal-Halifax and the Mediterranean, while individually securing transport between Montreal-Halifax and Northern Europe, and Northern Europe and the Mediterranean respectively. The Montreal service additionally interacts with a service between Europe and Asia, which is partly illustrated.

### B.1.1 Modelling the Liner Shipping Network Design Problem (LSNDP)

The Liner Shipping Network Design Problem (LSNDP) aims to optimize the design of the networks to minimize cost, while satisfying customer service requirements and operational constraints. The mathematical formulation of the LSNDP may be very rich as seen in (Løfstedt et al. [2010]), where a compact formulation along with an extensive set of service requirements and network restrictions is presented. A rich formulation like (Løfstedt et al. [2010]) serves as a description of the LSNDP domain, but is not computa-

tionally tractable as the number of feasible services is exponential in the number of ports. Therefore, a formulation of the LSNDP is typically restricted to an interpretation of the domain along with the core costs and constraint structures of the problem. The LSNDP has been modelled as a rich Vehicle Routing Problem (VRP) (Baldacci et al. [2010]) for instance, where transhipments are not allowed and vessels can be assumed to return empty to a single main port of a voyage in (Fagerholt [2004]), (Karlaftis et al. [2009]). The structure is applicable for regional liner shippers referred to as *feeder* services as opposed to global liner shipping in focus in the present paper. Models where the LSNDP is considered as a specialized capacitated network design problem with multiple commodities are found in (Reinhardt and Kallehauge [2007]), (Agarwal and Ergun [2008]), (Alvarez [2009]), (Plum [2010]). The network design problem is complicated by the network consisting of disjoint cycles representing container vessel routes as opposed to individual links. The models handle transhipments although transhipment cost is not included in (Agarwal and Ergun [2008]). The vessels are not required to be empty at any time. The works of (Agarwal and Ergun [2008], Alvarez [2009]) identify a two tier structure of constraint blocks: the first deciding the rotations of a single or a collection of vessels resulting in a capacitated network and the second regarding a standard multicommodity flow problem with a dense commodity matrix. The cost structure of LSNDP places vessel related costs in the first tier and cargo handling cost and revenue in the second tier. The work of (Plum [2010]) has identified two main issues with solving the LSNDP as a specialized capacitated network design problem:

1. Economy of scale on vessels and the division of cost and revenue on the two tiers results in highly fractional LP solutions.

2. The degeneracy of the multicommodity flow problem results in weak LP bounds.

Furthermore, it is well known that the linear multicommodity flow problem and hence capacitated network design problems do not scale well with the number of distinct commodities. Computational results for existing models confirm the hardness of this problem and the scalability issues, struggling to solve instances with 10-15 ports and 50-100 commodities.

The model presented in this paper has a single tier and combines revenue with total cost in the service generation problem. The motivation is to ensure efficient capacity utilization of vessels and avoid highly fractional LP solutions. Service generation is based on pick-up-and-delivery of cargoes transported entirely or partly on the service. The cost of a service reflects asset, operational and port call costs of the vessels on the service, along with the cargo handling cost and revenue of collected cargo on the service. The cargo handling cost includes load, unload and transhipment costs. The

model is inspired by the Pick-up-and-Delivery VRP problem, but is considerably more complex as we allow transhipments on non-simple cyclic routes, where the vessel is not required to be empty at any point in time.

The degeneracy of the multicommodity flow problem is mitigated both by modelling the flow as assignments to services as opposed to the traditional multicommodity flow formulation, but also by exploiting the liner shipping concept of *trade* lanes to aggregate the number of distinct commodities to a minimum. Trade lanes are based on the geographic distances within a set of ports and their potential to import/export to another region.

Maritime shipping produces an estimated 2.7% of the worlds $CO_2$ emission, whereof 25% is accounted to container vessels according to the (WorldShippingCouncil [2010]). The value proposition of liner shipping companies has focus on the environmental impact of their operation and the concept of *slow steaming* has become a standard for some liner shipping companies (List [2010]). (Cariou [2011]) estimate that the emissions have decreased by 11 % since 2008 by slow steaming alone. Breaking down the cost of a service to each vessel (Stopford [1998]) state that 35-50% of the cost is for fuel (bunker) whereas capital cost accounts for 30-45%, OPEX (crew, maintenance and insurance) accounts for 6-17% and port cost for 9-14%. Slow steaming minimizes the fuel cost, but comes at an asset cost of additional vessels deployed to maintain weekly frequency (Notteboom and Vernimmen [2009]). Slow steaming is not always an option as some cargo may have crucial transit times. Current models of LSNDP assumes fixed speed on a service. The model of (Alvarez [2009]) explicitly aims at minimizing the fuel cost and consumption in the network by varying the speed of services in the model. (Løfstedt et al. [2010]), (Notteboom and Vernimmen [2009]) and (Fagerholt et al. [2010]) state that the speed on a service is variable on each individual voyage between two ports and as the fuel consumption is a cubic function of speed (Stopford [1998]) the cost calculated on an average fixed speed on a roundtrip is an approximation. As a result the actual fuel consumption of a service cannot be estimated until the schedule is fixed. Tramp shipping often model their routing and scheduling as rich Pick-up-and-Delivery VRP problems with Time Windows (Fagerholt and Lindstad [2007], J. E. Korsvik and Laporte [2010]). (Fagerholt et al. [2010]) is the first article within tramp shipping with variable speed between each port pair in the routing. The optimization of speed and hence minimizing the fuel consumption and environmental impact is driven by the time windows and the optional revenue of spot cargoes. (Fagerholt et al. [2010], I. Norstad and Laporte [2010]) report significant improvements in solutions using variable speed. Minimizing the fuel consumption of the network can be a post optimization regarding speed of the liner shipping network, when deciding on the schedule in terms of berthing windows or the transit time of individual cargo routings. The path based model presented in this paper assumes a fixed speed for each vessel class and in the dynamic programming algorithm

the number of vessels deployed to a service is ceiled in order to ensure that a weekly frequency can be maintained on each service.

The path based model is inspired by operations research techniques within the airline industry, where the optimization is divided into faces. Therefore, a solution to the path based model is a generic capacitated network of cyclic services based on a weekly frequency of port calls. The generic network is transformed into an actual network by making an actual schedule, deploying vessels and deciding on the speed of the individual voyages and actual flow of all distinct commodities. The slow steaming speed of a vessel is 12 knots and depending on size and age a vessel has a maximal speed of 18 to 25 knots. If the fixed speed is chosen 30-40% above slow steaming speed for each vessel the ceiling of the number of vessels will allow post optimization of the schedule to achieve an energy efficient network with focus on slow steaming, while ensuring the transit time of products. The generic network allows for a green liner shipping network, while at the same time enabling scalability due to a more general description of the network.

### B.1.2 Demand Aggregation

In models of the LSNDP using a specialized capacitated network design formulation the second tier is a standard multicommodity flow problem. The work of (Alvarez [2009]) identifies solving the multicommodity flow problem as prohibitive for larger problem instances due to the large number of commodities considered. The model of (Alvarez [2009]) is aggregating the flow combining it by destination, giving a smaller model to solve. This could result in worse LP bounds as identified in (Croxton et al. [2007]), as the LSNDP will have a concave cost function, due to the economies of scales of deploying larger vessels, and high start up costs, as at least one vessel must be deployed.

A contribution of this paper is to formulate a model that considers aggregated aspects of the demand instead of specific origin-destination (*o*-*d*) pairs. This is motivated by the *trade*-centric view of liner-shipping present in the liner shipping industry instead of the *o*-*d*-centric view considered in the literature. As seen in Figure B.1, the o-d demand from Halifax to Rotterdam could be considered, but in practice it will be hard to estimate such a specific demand. More realistically one could estimate the volume of exports from Halifax to Northern Europe and reversely the volume of imports from East Coast Canada to Rotterdam (or exports from Mediterranean to Halifax as in Figure B.2). Each commodity $k \in K$ will then be characterized by a volume $d^{XY}$ from a region $X$ to a region $Y$ i.e. East Coast Canada or Northern Europe as seen in Figure B.1 on the vessels in deep sea. Each set of $X$,$Y$ will symbolize a *trade*. Each port $p \in X$ will also have an export and import in the trade: $d^{pY}, d^{Xp}$, where $\sum_{p \in X} d^{pY} = \sum_{p \in Y} d^{Xp}$ as seen in Figure B.1 on the vessels in a region. In effect a port as Halifax will be ensured

a volume of export to Mediterranean ports and each of these will be insured a volume of imports from East Coast Canadian ports, without specifying the concrete origin-destination pairs. Note the difference in aggregation approach, compared with the models of (Croxton et al. [2007]), as we are now aggregating by *trade* origin-region to destination-region, instead of aggregation by destination port. This should give the benefit of fewer variables due to the aggregation, while we still have quite tight LP-relaxations.

The aggregation of demand may be more or less fine grained giving the ability to "zoom" according to the definition of ports, regions and trade lanes, enabling both detailed networks for a smaller region and coarse network designs for a larger set of ports that may be refined by subsequent optimization methods. We foresee a computational tractability trade-off between the number of ports and the number of distinct commodities when defining regions for ports.

In the following we will present a path-based formulation of the LNSDP and a column generation approach generating capacitated cyclic rotations with assigned flow. We will outline a dynamic programming algorithm to solve the pricing problem.

## B.2 Service Based Model

In the following we introduce a model based on a combination of feasible services for each vessel class, into a generic liner shipping network solution. The service based model is based on a Dantzig-Wolfe decomposition of the model in (Løfstedt et al. [2010]). Let $S_v$ denote the set of feasible services for a vessel class $v \in V$ and let $S = \cup_{v \in V} S_v$. Let $\alpha_{kps}^{XY}$ and $\beta_{kps}^{XY}$ be the amount of respectively load and unload of containers from region $X$ to region $Y$ on the $k$'th visit to port $p$ on service $s \in S$. We assume that $\alpha_{kps}^{XY} = \beta_{kps}^{XY} = 0, \forall p \notin X \cup Y \cup G^{XY}$, where $G^{XY}$ is the set of ports where transhipments is allowed for *trade* $XY$. Let $M_p$ be the maximal number of port visits to port $p$ for each service. Furthermore, let $\gamma_{pq}$ equal the number of times the service sails between ports $p \in P$ and $q \in P$. The move cost in a port $p$ for a *trade* $XY \in K$ consist of the unload cost $u_p^{XY}$ and load cost $l_p^{XY}$. For ports $p \in X$ the transhipment cost is included in the unload cost and the revenue is $r_p^{XY}$. For ports $p \in P \setminus X$ the transhipment cost is included in the load cost. Each vessel of vessel type $v \in V$ has costs $c_v$ for fuel-, crew- and depreciation of vessel value or time-charter-costs per week. The cost of vessel type $v$ calling a port $q$ is $c_q^v$. The number of vessels used by the service is the round trip distance of the service divided by $W_d^v$, the weekly distance covered by vessel type $v$ at the predefined speed. This value is ceiled to ensure the vessels can complete the round trip at the predefined speed. The number of vessels used by the service is given as $n_s = \left\lceil \sum_{p \in P} \sum_{q \in P} \frac{d_{pq} \gamma_{pqs}}{W_d^v} \right\rceil$. The cost of a service $s \in S$ is given as:

$$
\begin{aligned}
c_s = & \sum_{XY \in K} \sum_{p \in X} \sum_{k \in M_p} r_p^{XY} (\alpha_{kps}^{XY} - \beta_{kps}^{XY}) \\
& - \sum_{XY \in K} \sum_{p \in P} \sum_{k \in M_p} (l_p^{XY} \alpha_{kps}^{XY} + u_p^{XY} \beta_{kps}^{XY}) \\
& - c_v n_s - \sum_{p \in P} \sum_{q \in P} c_q^v \gamma_{pqs}
\end{aligned}
$$

The model based on services is as follows:

$$\max \sum_{s \in S} c_s \lambda_s \tag{B.1}$$

$$\texttt{s.t } 0 \leq \sum_{s \in S} \sum_{k \in M_p} (\alpha_{pks}^{XY} - \beta_{pks}^{XY}) \lambda_s \leq d^{pY}$$

$$\forall XY \in K, \forall p \in X \tag{B.2}$$

$$0 \geq \sum_{s \in S} \sum_{k \in M_p} (\alpha_{pks}^{XY} - \beta_{pks}^{XY}) \lambda_s \geq -d^{Xp}$$

$$\forall XY \in K, \forall p \in Y \tag{B.3}$$

$$\sum_{s \in S} \sum_{k \in M_p} (\alpha_{pks}^{XY} - \beta_{pks}^{XY}) \lambda_s = 0$$

$$\forall p \in G^{XY}, \forall XY \in K \tag{B.4}$$

$$\sum_{s \in S} \sum_{p \in X \cup Y} \sum_{k \in M_p} (\alpha_{pks}^{XY} - \beta_{pks}^{XY}) \lambda_s = 0$$

$$\forall XY \in K \tag{B.5}$$

$$\sum_{s \in S_v} n_s \lambda_s \leq |v| \qquad \forall v \in V \tag{B.6}$$

$$\lambda_s \in \{0, 1\} \qquad \forall s \in S \tag{B.7}$$

The objective (B.1) maximizes the profit, constraints (B.2) and (B.3) ensure that the difference between what is loaded and unloaded (unloaded and loaded) by all services in a port is positive and less than the export capacity (import capacity) of the port for the given *trade*. Constraints (B.4) ensure that the amount of containers loaded equals the amount of containers unloaded in a transhipment port and constraints (B.5) ensure that all containers loaded are unloaded for each *trade*. Constraints (B.6) ensure that the number of available vessels for each vessel class is not exceeded and the binary domain on the variables is defined by (B.7).

The key issue with the service based model is that the set of feasible services $S$ can be exponential in the number of ports. Therefore it cannot be expected to solve instances of significant size. To overcome this issue we propose to write up the model gradually using delayed column generation

and then solve the problem through Branch-and-Cut-and-Price. Branching is done by imposing a limit on the number of times an arc can be used by a given vessel class. When possible an enumeration technique similar to the one used within CVRP (Baldacci et al. [2008]) will be used. The upper bound needed will be obtained using heuristics adapted from (Løfstedt et al. [2010]).

### B.2.1 Pricing Problem

The pricing problem returns a column to the master problem which represents a service. Each column consists of a load and an unload pattern, which implicitly defines a non-simple cycle starting and ending at the same port $p \in P^v$ deploying $n_s$ vessels to maintain weekly frequency at the fixed speed enforced on the service pattern. The pricing problem is to find a non-simple cycle $\sigma$ centered around any starting node $p_s$ with associated loads and unloads such that the capacity of the vessel class ($C_v$) is not exceeded at any port $p$, the distance of the schedule with a weekly frequency is feasible for the vessel class $v$ and no port $p$ is visited more than $M_p$ times $\forall p \in P$. The above problem has a similar structure to the pricing problems that arise in the context of Vehicle Routing (Resource Constrained Shortest Path problems (see Irnich and Desaulniers [2005]) usually solved by label setting algorithms.

#### Reduced cost

To use delayed column generation we start by considering the reduced cost of a column (service). For each $XY \in K$ a port $p \in P$ is present in at most one of the constraints (B.2) to (B.4). Let $\omega_p^{XY}, \forall XY \in K, \forall p \in X \cup Y \cup G^{XY}$ denote the duals from (B.2) to (B.4). Let $\delta^{XY}$ be the dual variables of constraints (B.5) and $\pi^v$ are the duals of constraints (B.6).

For each vessel class $v \in V$ the reduced cost of a service(column) $s \in S_v$

$$
\begin{aligned}
\hat{c}_s =& c_s - \sum_{XY \in K} \sum_{p \in X \cup Y \cup G^{XY}} \sum_{k \in M_p} \omega_p^{XY} (\alpha_{kps}^{XY} - \beta_{kps}^{XY}) \\
& - \sum_{XY \in K} \sum_{p \in X \cup Y} \sum_{k \in M_p} \delta^{XY} (\alpha_{kps}^{XY} - \beta_{kps}^{XY}) - \pi_v n_s \\
=& \sum_{XY \in K} \sum_{p \in X} \sum_{k \in M_p} r_p^{XY} (\alpha_{kps}^{XY} - \beta_{kps}^{XY}) \\
& - \sum_{XY \in K} \sum_{p \in P} \sum_{k \in M_p} l_p^{XY} \alpha_{kps}^{XY} \\
& - \sum_{XY \in K} \sum_{p \in X \cup Y \cup G^{XY}} \sum_{k \in M_p} \omega_p^{XY} \alpha_{kps}^{XY} \\
& - \sum_{XY \in K} \sum_{p \in X \cup Y} \sum_{k \in M_p} \delta^{XY} \alpha_{kps}^{XY} \\
& - \sum_{XY \in K} \sum_{p \in P} \sum_{k \in M_p} u_p^{XY} \beta_{kps}^{XY} \\
& + \sum_{XY \in K} \sum_{p \in X \cup Y \cup G^{XY}} \sum_{k \in M_p} \omega_p^{XY} \beta_{kps}^{XY} \\
& + \sum_{XY \in K} \sum_{p \in X \cup Y} \sum_{k \in M_p} \delta^{XY} \beta_{kps}^{XY} \\
& - c_v n_s - \sum_{p \in P} \sum_{q \in P} c_q^v \gamma_{pqs} - \pi_v n_s \\
=& \sum_{XY \in K} \sum_{p \in X} \sum_{k \in M_p} (r_p^{XY} - l_p^{XY} - \omega_p^{XY} - \delta^{XY}) \alpha_{kps}^{XY} + \\
& \sum_{XY \in K} \sum_{p \in X} \sum_{k \in M_p} (-r_p^{XY} - u_p^{XY} + \omega_p^{XY} + \delta^{XY}) \beta_{kps}^{XY} + \\
& \sum_{XY \in K} \sum_{p \in Y} \sum_{k \in M_p} (-l_p^{XY} - \omega_p^{XY} - \delta^{XY}) \alpha_{kps}^{XY} + \\
& \sum_{XY \in K} \sum_{p \in Y} \sum_{k \in M_p} (-u_p^{XY} + \omega_p^{XY} + \delta^{XY}) \beta_{kps}^{XY} + \\
& \sum_{XY \in K} \sum_{p \in G^{XY}} \sum_{k \in M_p} (-l_p^{XY} - \omega_p^{XY}) \alpha_{kps}^{XY} + \\
& \sum_{XY \in K} \sum_{p \in G^{XY}} \sum_{k \in M_p} (-u_p^{XY} + \omega_p^{XY}) \beta_{kps}^{XY} \\
& - (\pi_v + c_v) n_s - \sum_{p \in P} \sum_{q \in P} c_q^v \gamma_{pqs}
\end{aligned}
$$

The reduced cost can be rewritten as a cost connected to loading, unloading, and sailing in terms of the number of vessels deployed and the cumulative

port call cost:

$$\hat{l}_p^{XY} = \begin{cases} r_p^{XY} - l_p^{XY} - \omega_p^{XY} - \delta^{XY} & \forall p \in X \\ -l_p^{XY} - \omega_p^{XY} - \delta^{XY} & \forall p \in Y \\ -l_p^{XY} - \omega_p^{XY} & \forall p \in G^{XY} \end{cases}$$

$$\hat{u}_p^{XY} = \begin{cases} -r_p^{XY} - u_p^{XY} + \omega_p^{XY} + \delta^{XY} & \forall p \in X \\ -u_p^{XY} + \omega_p^{XY} + \delta^{XY} & \forall p \in Y \\ -u_p^{XY} + \omega_p^{XY} \forall p \in G^{XY} & \end{cases}$$

Finally, the port call cost $c_q^v$ is paid upon each sailing/extension onto a new port $p \in P$ and the cost $\hat{c_v} = \pi_v + c_v$ is inferred each time the distance of $W_d^v$ is traveled.

### Graph topology and label setting algorithm for LSNDP

The $|V|$ pricing problems can be formulated as the following graph problem. Given a directed graph $G^v = (N^v, A^v)$ where the node set $N^v = P^v \cup L^v \cup U^v$. $P^v$ is the set of ports $\in P$ compatible with vesselclass $v$, $L^v = \bigcup_{w \in P^v} L_w$ the set of load nodes, where the sets $L_w = \{\rho_w^{XY} | \forall XY \in K, w \in X \vee Y \vee G^{XY}\}$ represents all possible loads at port $w$, $U^v = \bigcup_{w \in P^v} U_w$ is the set of unload nodes, where the sets $U_w = \{\mu_w^{XY} | \forall XY \in K, w \in X \vee Y \vee G^{XY}\}$ represents all possible unloads at port $w$. In order to correctly identify transhipments and unloads of a *trade* each demand $XY \in K$ is associated with a set of load nodes $L^{XY} \subseteq L^v$ and a set of unload nodes $U^{XY} \subseteq U^v$, where $L^{XY} = \{\rho_w^{XY} | w \in X \cup Y \cup G^{XY}\}$ and $U^{XY} = \{\mu_w^{XY} | w \in X \cup Y \cup G^{XY}\}$.

The arc set $A^v = A_s \cup A_u \cup A_l$. Define the function $h : U^v \cup L^v \rightarrow P^v, L_q \mapsto q, U_q \mapsto q$ for mapping between the load and unload nodes and the actual port $q \in p^v$ of the (un)load. The set of sail arcs is defined as follows $A_s = \{(i,j) | i \in L^v \cup U^v, j \in P^v \setminus \{h(i)\}\}$ , the set of unload arcs $A_u = \{(i,j) | i \in P^v, j \in U_i\} \cup \{(i,j) | i \in U^v, j \in U_{h(i)}\}$ and the set of load arcs $A_l = \{(i,j) | i \in P^v, j \in L_i\} \cup \{(i,j) | i \in U^v = \mu_{h(i)}^{XY}, j \in L_{h(i)} \setminus \{\rho_{h(i)}^{XY}\}\} \cup \{(i,j) | i \in L^v, j \in L_{h(i)}\}$. The graph topology is illustrated in figure B.3. The distance of an arc depends on the arc type: $d_{ij} = \begin{cases} d_{h(i)j} & (i,j) \in A_s \\ 0 & (i,j) \in A_l \cup A_u \end{cases}$

In a label setting algorithm a label $E_i$ is associated with a node $i$ and represents a (partial) path with a (reduced) cost $C$ of the service and a number of resources $\theta$ accumulated along the path. A resource may be associated with lower and upper bounds often referred to as a *resource window*. The proposed algorithm differs significantly from the Elementary Shortest Path Problem with Resource Constraints (ESPPRC) known from VRP:

- The path is not elementary as $M_p \geq 1$.

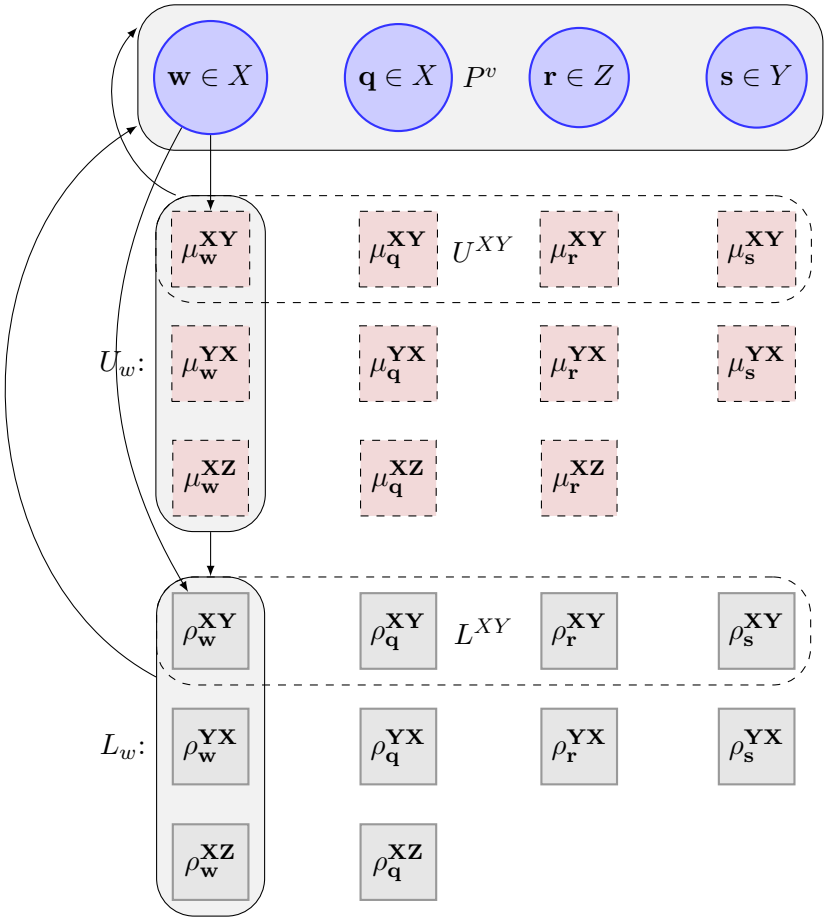



Figure B.3: A network representation of a graph associated with the label setting algorithm. The set of port call nodes $P^v$ is a clique. For port $w \in P^v$ the sets $U_w, L_w$ are illustrated. They represent possible loads and unloads at port $w$. The sets $U_w, L_w$ are cliques. A path in the network will follow sequences of $n \in P_v \to U_n \to L_n \to m \in P_v$. It is possible to only unload or load. The load set of a port $w$ is not connected to the unload set of $w$. Each trade $XY \in K$ is associated with a loadset $L^{XY}$ and an unloadset $U^{XY}$ as illustrated.

- The path represents a cycle, $\sigma$.
- It is a longest cycle problem as the reduced $\hat{c_s} \geq 0$.
- We do not have a designated starting node and hence will have to start the algorithm $\forall p \in P^v$.
- The ability to perform a load on the partial path, which can be unloaded at a previous node of the cycle $\sigma$. A second pass of all ports in the cycle $\sigma$ must be performed only allowing the unload extension function to check for load balance.
- The route is combined with a loading/unloading pattern not unlike the labelling algorithm for the SDVRPTW in Desaulniers [2010].

In the label setting algorithm for LSNDP a label $E$ contains the following information:

- Current port, $p_c$
- Start port, $p_s$
- (reduced) cost, $t$
- Accumulated distance, $d_s$
- The load of each *trade*, $F^{XY} \quad \forall XY \in K$
- Current load, $F_c = \sum_{XY \in K} F^{XY}$
- Visit number, $k_p \quad \forall p \in P^v$

The resources are $d_s, (F^{XY})_{XY \in K}, F_c, (k_p)_{p \in P^v}$ i.e. we have $2+|K|+|p^v|$ ressources. The extension function of the distance is defined as $e^d_{(ij)}(E_i) = d(E_i) + d_{ij}$. The feasibility and ressource consumption of extending label $E_i$ along an arc depends on the arc type:

- *Case 1: extending along a sail arc* $(i,j) \in A_s$
  A feasible extension of label $E_i$ to node $j$ along a sail arc $(i,j) \in A_s$ must satisfy the following conditions:

$$\left\lceil \frac{e^d_{(ij)}(E_i)}{W^v_d} \right\rceil \leq |v| \tag{B.8}$$

$$k^i_j + 1 \leq M_j \tag{B.9}$$

  (B.8) ensures the feasibility of the number of vessels deployed to the service and (B.9) ensures the number of port calls to port $j$ does not exceed $M_j$. If the extension is feasible a new label $E_j$ is created. Define

$\varpi = \left\lceil \frac{e^d_{(ij)}(E_i)}{W^v_d} \right\rceil - \left\lceil \frac{d(E_i)}{W^v_d} \right\rceil$. $\varpi$ expresses whether the label extension will require an additional vessel on the service to maintain weekly frequency. The following extension functions are applied to create label $E_j$: $p^j_c = j, p^j_s = p^i_s, t^j = t^i - c^v_j - \hat{c_v} \cdot \varpi, d = e^d_{(ij)}(E_i), F^j_C = F^i_C, F^{XY}_j = F^{XY}_i, k^j_j = k^i_j + 1, k^j_p = k^i_p \quad \forall p \in P^v \setminus \{j\})$

- *Case 2: extending along an unload arc* $(i,j) \in A_u | j = \mu^{XY}_p$
  A feasible extension of label $E_i$ to node $j$ along unload arc $(i,j) \in A_u$ must satisfy the following conditions:

$$F^{XY} > 0 \tag{B.10}$$

  where (B.10) ensures that the commodity $XY$ is currently loaded on the vessel i.e. that a previous visit to a node in $L^{XY}$ has been performed. If the extension is feasible a new label $E_j$ is created using the extension functions:
  $p^j_c = h(j), p^j_s = p^i_s, t^j = t^i + \hat{u}^{XY}_p \cdot max\{d^{XY}, F^{XY}_i\}, d = e^d_{(ij)}(E_i), F^j_C = F^i_C - max\{d^{XY}, F^{XY}_i\}, F^{XY}_j = F^{XY}_i - max\{d^{XY}, F^{XY}_i\}, F^{ZW}_j = F^{ZW}_i \forall ZW \in K \setminus \{XY\}, k^j_p = k^i_p \quad \forall p \in P^v)$

- *Case 3: extending along a load arc* $(i,j) \in A_l | j = \rho^{XY}_p$
  A feasible extension of label $E_i$ to node $j$ along a load arc $(i,j) \in A_l$ must satisfy the following conditions:

$$F^i_c < C_v \tag{B.11}$$

  (B.11) ensures that the vessel has excess capacity for loading.

  If the extension is feasible a new label $E_j$ is created with the following extension function: $p^j_c = h(j), p^j_s = p^i_s, t^j = t^i + \hat{l} \cdot max\{d^{XY}, C_v - F^i_C\}, d = e^d_{(ij)}(E_i), F^j_C = F^i_C + max\{d^{XY}, C_v - F^i_C\}, F^{XY}_j = F^{XY}_i + max\{d^{XY}, C_v - F^i_C)\}, F^{ZW}_j = F^{ZW}_i \forall ZW \in K \setminus \{XY\}, k^j_p = k^i_p \quad \forall p \in P^v)$

A state is feasible when the start node is reached ($p_c = p_s$) and the containers are balanced for all trades( $F^{XY} = 0 \ \forall XY \in K$) by applying unload extensions to the cycle starting from $p_s$ ending in $p_s$. To obtain the solution to a service the auxiliary data of what has actually been loaded and unloaded has to be stored and a mapping from $L$ to $\alpha$ and from $U$ to $\beta$ creates the column entries for (un)load in the master problem. For an exact solution to the pricing problem the service with the best reduced cost (max $\hat{c_s}$) is added to the master problem. However, the label setting algorithm may find several services where the cost $t$ is greater than 0 and add several columns in an iteration to accelerate convergence of the column generation algorithm.

### B.2.2 Complexity

Let $T$ denote an upper bound on the distance. The running time of the algorithm can be shown to be $O((T|P|C^{|K|}\prod_{p\in X} d^{pY}\prod_{p\in G^{XY}} C)^2)$. Increasing the number of *trades* and the number of transhipment ports will increase the number of states in the Dynamic Programming algorithm. To solve practical problem instances it is therefore important to make a careful choice of the *trades* and the ports, where transhipment is allowed.

In CVRP a pseudo polynomial relaxation is used when solving the strongly NP-hard pricing problem (Baldacci et al. [2008]) to reduce the practical running time of the algorithm. The method has proven to be very powerful and we therefore suggest a pseudo polynomial relaxation of our pricing problem. This relaxation can be obtained as follows: Each port is assigned the minimal load and unload cost and the bounds on the load is removed. In each port the number of different states will then be limited to $T|P||C|$ and a running time of $O((T|P|^2|C|))$ can be obtained.

As in other column generation algorithms we will not solve the pricing problem to optimality in each iteration but will stop once a sufficient amount of columns with positive reduced cost is found. An easy way to do this is to run the dynamic programming algorithm using a greedy variant adding any reduced cost column instead of the best reduced cost column.

## B.3 Conclusion

We have presented a new model for LSNDP and presented a solution approach using column generation. Among the benefits of the proposed model is a novel view of demands in liner-shipping, which are considered on a *trade* basis. This has the advantage of both being intrinsic to the liner shipping business, giving a natural understanding, and requiring fewer variables. Additionally the proposed subproblem is related to the pricing problems in VRP where Branch-and-Cut-and-Price has been used with great success. We have shown that a pseudo polynomial relaxation can be used as bounding to solve the pricing problem in combination with heuristics and other techniques that have been effective in solving VRP problems. This encourages us to believe that the method scales well to larger instances. In the VRP context resource limitations have proven to be effective for the dynamic programming algorithms in reducing the state space. Therefore, further work with richer formulations of LSNDP, considering aspects as transit time limits on paths, draught limits in ports and other operational constraints from liner shipping might tighten the pricing problems. This will help us scale to larger instances while adding real life complexity to the model. At the time of the conference we aim to present preliminary computational results for the dynamic programming algorithm based on data from the benchmark paper of (Løfstedt et al. [2010]).

# Bibliography

R. Agarwal and O. Ergun. Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science*, 42(2):175–196, 2008. Cited By (since 1996): 6.

J. F. Alvarez. Joint routing and deployment of a fleet of container vessels. *Maritime Economics and Logistics*, 11(2):186–208, 2009. Cited By (since 1996): 1.

R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115(2):351–385, 10 2008.

R. Baldacci, E. Bartolini, A. Mingozzi, and R. Roberti. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science*, 7(3):229–268, 2010.

Pierre Cariou. Is slow steaming a sustainable means of reducing co2 emissions from container shipping? *Transportation Research Part D: Transport and Environment*, 16(3):260 – 264, 2011. ISSN 1361-9209. doi: DOI: 10.1016/j.trd.2010.12.005.

Keely L. Croxton, Bernard Gendron, and Thomas L. Magnanti. Variable disaggregation in network flow problems with piecewise linear costs. *Operations research*, 55(1):146–157, 2007.

Guy Desaulniers. Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations Research*, 58(1):179–192, January 1 2010.

K. Fagerholt. Designing optimal routes in a liner shipping problem. *Maritime Policy and Management*, 31(4):259–268, 2004. Cited By (since 1996): 13.

K. Fagerholt and H. Lindstad. Turborouter: An interactive optimisation-based decision support system for ship routing and scheduling. *Maritime Economics and Logistics*, 9:214–233, 2007.

K. Fagerholt, G. Laporte, and I. Norstad. Reducing fuel emissions by optimizing speed on shipping routes. *Journal of the Operational Research Society*, 61(3):523–529, 2010.

K. Fagerholt I. Norstad and G. Laporte. Tramp ship routing and scheduling with speed optimization. *Transportation Research Part C: Emerging Technologies*, 2010. In Press.

S. Irnich and G. Desaulniers. *Shortest Path Problems with Resource Constraints*, chapter 2 in Column Generation (eds. G. Desaulniers and Jacques Desrosiers and M.M. Solomon), pages 33–65. GERAD 25th Anniversary Series. Springer, 2005.

K. Fagerholt J. E. Korsvik and G. Laporte. A large neighbourhood search heuristic for ship routing and scheduling with split loads. *Computers & Operations Research*, 2010. Accepted for publication.

M. G. Karlaftis, K. Kepaptsoglou, and E. Sambracos. Containership routing with time deadlines and simultaneous deliveries and pick-ups. *Transportation Research Part E: Logistics and Transportation Review*, 45(1): 210–221, 2009. Cited By (since 1996): 4.

Lloyd's List. Maersk line chief operating officer, morten engelstoft: Slow steaming is here to stay. Lloyd's List 2010, 2010.

B. Løfstedt, J. F. Alvarez, C.E.M. Plum, D. Pisinger, and M.M. Sigurd. An integer programming model and benchmark suite for liner shipping network design. Technical Report 19, Technical University of Denmark, 2010.

T.E. Notteboom and B Vernimmen. The effect of high fuel costs on liner service configuration in container shipping. *Journal of Transport Geography*, 17:325–337, 2009.

C.E.M. Plum. The linearized simultaneous string-design and cargo-routing problem. EURO 2010 conference, 07 2010.

L. B. Reinhardt and B. Kallehauge. Network design for container shipping using cutting planes. 12 2007. Conference on Maritime & Intermodal Logistics, Singapore.

M. Stopford. *Maritime Economics*. Routledge, 3 edition, 1998.

WorldShippingCouncil. Liner shipping and co2 emissions policy. www.shippingandco2.org/ LinerShippingandCO2EmissionsPolicySeptember.pdf, 2010.

# Appendix C Partial Path Column Generation for the Vehicle Routing Problem with Time Windows

Bjørn Petersen Mads Kehlet Jepsen
*DIKU Department of Computer Science, University of Copenhagen*

April 2009

## Abstract

This paper presents a column generation algorithm for the Vehicle Routing Problem with Time Windows (VRPTW). Traditionally, column generation models of the VRPTW have consisted of a Set Partitioning master problem with each column representing a route, i.e., a resource feasible path starting and ending at the depot. Elementary routes (no customer visited more than once) have shown superior results on difficult instances (less restrictive capacity and time windows). However, the pricing problems do not scale well when the number of feasible routes increases, i.e., when a route may contain a large number of customers. We suggest to relax that 'each column is a route' into 'each column is a part of the giant tour'; a so-called partial path, i.e., not necessarily starting and ending in the depot. This way, the length of the partial path can be bounded and a better control of the size of the solution space for the pricing problem can be obtained.

## C.1 Introduction

The VRPTW can be described as follows: A set of customers, each with a demand, needs to be serviced by a number of vehicles all starting and ending at a central depot. Each customer must be visited exactly once within a given time window and the capacity of the vehicles may not be exceeded. The objective is to service all customers traveling the least possible distance. In this paper we consider a homogeneous fleet, i.e., all vehicles are identical.

The standard Dantzig-Wolfe decomposition of the arc flow formulation of the VRPTW is to split the problem into a master problem (a Set Partitioning Problem) and a pricing problem (an Elementary Shortest Path Problem with Resource Constraints (ESPPRC), where capacity and time are the constrained resources). A restricted master problem can be solved with delayed column generation and embedded in a branch-and-bound framework to ensure integrality. Applying cutting planes either in the master or the pricing problem leads to a Branch-and-Cut-and-Price algorithm (BCP). Dror [5] showed that the ESPPRC (with time and capacity) is strongly $\mathcal{NP}$-hard.

We propose a decomposition approach based on the generation of partial paths and the concatenation of these. In the bounded partial path decomposition approach the main idea is to limit the solution space of the pricing problem by bounding some resource, e.g., the number of nodes on a path. The master problem combines a known number of these bounded partial paths to ensure all customers are visited.

The paper is organized as follows: In Section C.2 an overview of the Dantzig-Wolfe decomposition of the VRPTW is given and it is described how to calculate the reduced cost of columns when column generation is used. Section C.5 concludes on the model.

## C.2 The Vehicle Routing Problem with Time Windows

The VRPTW can formally be stated as: Given a graph $G(V,A)$ with nodes $V$ and arcs $A$, a set $R$ of resources ($R = \{\text{load}, \text{time}\}$) where each resource $r \in R$ has a lower bound $a_i^r$ and an upper bound $b_i^r$ for all $i \in V$ and a positive consumption $\tau_{ij}^r$ when using arc $(i,j) \in A$, find a set of routes starting and ending at the depot node $0 \in V$ satisfying all resource limits, such that the cost is minimized and all customers $C = V \setminus \{0\}$ are visited.

**2-index formulation of the VRPTW** In the following let $c_{ij}$ be the cost of arc $(i,j) \in A$, $x_{ij}$ be the binary variable indicating the use of arc $(i,j) \in A$, and $T_{ij}^r$ be the consumption of resource $r \in R$ at the beginning of arc $(i,j) \in A$. Let $\delta^+(i)$ and $\delta^-(i)$ be the set of outgoing respectively ingoing arcs of node $i \in V$. The mathematical model of VRPTW adapted from Bard et al. [2] and Ascheuer et al. [1]:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{C.1}$$

$$\text{s.t.} \sum_{(i,j)\in\delta^+(i)} x_{ij} = 1 \qquad \forall i \in C \tag{C.2}$$

$$\sum_{(j,i)\in\delta^-(i)} x_{ji} = \sum_{(i,j)\in\delta^+(i)} x_{ij} \qquad \forall i \in V \tag{C.3}$$

$$\sum_{(j,i)\in\delta^-(i)} (T_{ji}^r + \tau_{ji}^r x_{ji}) \leq \sum_{(i,j)\in\delta^+(i)} T_{ij}^r \qquad \forall r \in R,\ \forall i \in C \tag{C.4}$$

$$a_i x_{ij} \leq T_{ij}^r \leq b_i x_{ij} \qquad \forall r \in R,\ \forall (i,j) \in A \tag{C.5}$$

$$T_{ij}^r \geq 0 \qquad \forall r \in R,\ \forall (i,j) \in A \tag{C.6}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{C.7}$$

The objective (C.1) sums up the cost of the used arcs. Constraints (C.2) ensure that each customer is visited exactly once, and (C.3) are the flow conservation constraints. Constraints (C.4) and (C.5) ensure the resource windows are satisfied. It is assumed that the bounds on the depot are always satisfied. Note, no sub-tours can be present since only one time stamp per arc exists and the travel times are positive.

## C.3 Bounded partial paths

A solution to the VRPTW: $v_0 \to c_1^1 \to \ldots \to c_{k_1}^1 \to v_0, v_0 \to c_1^2 \to \ldots \to c_{k_2}^2 \to v_0, \ldots, v_0 \to c_1^n \to \ldots \to c_{k_n}^n \to v_0$ can be represented by the giant-tour representation of Christofides and Eilon [3]:

$$v_0 \to c_1^1 \to \ldots \to c_{k_1}^1 \to v_0 \to c_1^2 \to \ldots \to c_{k_2}^2 \to v_0 \ldots \to v_0 \to c_1^n \to \ldots \to c_{k_n}^n \to v_0$$

which is one long path visiting all customers. The consumption of resources is reset each time the depot node is encountered.

The idea is to partition the problem so that the solution space of each part is smaller than the original problem. This is done by splitting the giant-tour into smaller segments by imposing an upper

limit on some resource, e.g., bounding the path length in the number of nodes. In the following the number of visited customers is considered the bounding resource, i.e., the number of visits to the non-depot node set $C$. Each segment represents a partial path of the giant-tour. With a fixed number of customers on each partial path, say $L$, a fixed number of partial paths, say $K$, is needed to ensure that all customers are visited, i.e., $L \cdot K \geq |C|$. The partial paths can start and end in any node in $V$ and can visit the depot several times. Example of a partial path:

$$c_1 \rightarrow c_2 \rightarrow v_0 \rightarrow c_3 \rightarrow v_0 \rightarrow c_4$$

Consider the graph $G'(V',A')$ consisting of a set of layers $K = \{1,\ldots,|K|\}$, each one representing $G$ for a partial path. Let $G^k$ be the sub graph of $G'$ representing layer $k$ with node set $V^k = \{(i,k) : i \in V\}$ for all $k \in K$ and arc set $A^k = \{(i,j,k) : (i,j) \in A\}$ for all $k \in K$. Let $A^* = \{(i,i,k) : (i,k) \in V^k \wedge (i,k+1) \in V^{k+1} \wedge k \in K\}$ be the set of interconnecting arcs, i.e., the arcs connecting a layer $k$ with the layer above $k$ namely layer $k+1$ for all $k \in K$ and all nodes $i \in V$ (layer $|K|+1$ is defined to be layer $1 \in K$ and layer 0 is defined to be layer $|K| \in K$). Let $V' = \bigcup_{k \in K} V^k$ and let $A' = \bigcup_{k \in K} A^k \cup A^*$. An illustration of $G'$ can be seen on Figure C.1. Note, that arc $(i,i,k)$ does not exist in $A^k$ and that arc $(i,j,k)$ with $i \neq j$ does exist in $A^*$, so all arcs $(i,j,k) \in A'$ can be uniquely indexed. With the length of a path defined as the number of customers on it, the problem is now to find partial paths of length at most $L$ in $|K|$ layers with $L \cdot |K| \geq |C| > L \cdot (|K|-1)$, so that each partial path $p$ ending in node $i \in V$ is met by another partial path $p'$ starting in $i$. All partial paths are combined while not visiting any customers more than once and satisfying all resource windows. A customer $c \in C$ is considered to be on a partial path $p$ if $c$ is visited on $p$ and is not the end node of $p$.

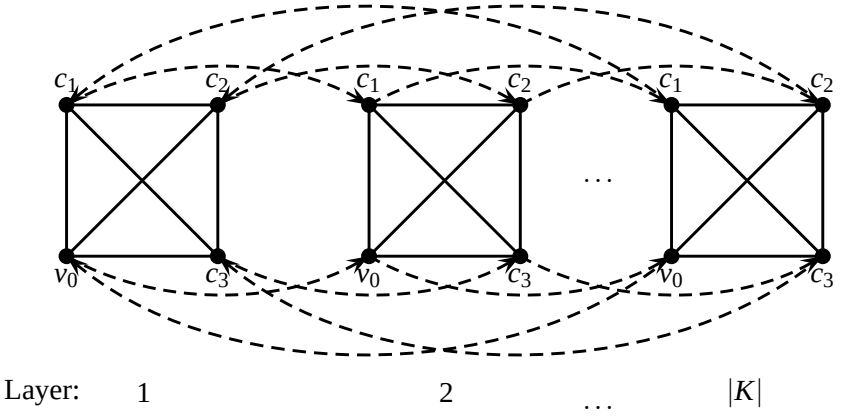



Figure C.1: Illustration of $G'$ with $|C| = 3$, $|K| = 3$, and $|L| = 1$. Edges (full-drawn) represent two arcs; one in each direction. Dashed lines are the interconnecting arcs $A^*$.

Let $L$ be the upper bound on the length of each partial path, and let $|C|$ be the length of the combined path (the giant-tour). Now, exactly $|K| = \lceil |C|/L \rceil$ partial paths are needed to make the combined path, since $L\lceil |C|/L \rceil \geq |C| > L(\lceil |C|/L \rceil - 1)$. Note that given a $|K|$, $L$ can be reduced to $L = \lceil |C|/|K| \rceil$.

**3-index formulation of the VRPTW** Let $x_{ij}^k$ be the variable indicating the use of arc $(i,j,k) \in A'$. Problem (C.1)–(C.7) is rewritten:

$$\min \sum_{k\in K}\sum_{(i,j)\in A} c_{ij}x_{ij}^k \tag{C.8}$$

$$\texttt{s.t.} \sum_{k\in K}\sum_{(i,j)\in\delta^+(i)} x_{ij}^k = 1 \qquad \forall i \in C \tag{C.9}$$

$$\sum_{(i,j)\in\delta^+(i)} x_{ij}^k \le 1 \qquad \forall k \in K,\ \forall i \in C \tag{C.10}$$

$$\sum_{k\in K}\left(x_{ii}^{k-1} + \sum_{(j,i)\in\delta^-(i)} x_{ji}^k\right) = \sum_{k\in K}\left(x_{ii}^k + \sum_{(i,j)\in\delta^+(i)} x_{ij}^k\right) \qquad \forall i \in V \tag{C.11}$$

$$x_{ii}^{k-1} + \sum_{(j,i)\in\delta^-(i)} x_{ji}^k = x_{ii}^k + \sum_{(i,j)\in\delta^+(i)} x_{ij}^k \qquad \forall k \in K,\ \forall i \in V \tag{C.12}$$

$$\sum_{k\in K}\sum_{i\in V} x_{ii}^k = K \tag{C.13}$$

$$\sum_{i\in C}\sum_{(i,j)\in A} x_{ij}^k \le L \qquad \forall k \in K \tag{C.14}$$

$$\sum_{k\in K}\sum_{(j,i)\in\delta^-(i)}\left(T_{ji}^{rk} + \tau_{ji}^r x_{ji}^k\right) \le \sum_{k\in K}\sum_{(i,j)\in\delta^+(i)} T_{ij}^{rk} \qquad \forall r \in R,\ \forall i \in C \tag{C.15}$$

$$\sum_{(j,i)\in\delta^-(i)}\left(T_{ji}^{rk} + \tau_{ji}^r x_{ji}^k\right) \le \sum_{(i,j)\in\delta^+(i)} T_{ij}^{rk} \qquad \forall r \in R,\ \forall k \in K,\ \forall i \in C \tag{C.16}$$

$$a_i \sum_{k\in K} x_{ij}^k \le \sum_{k\in K} T_{ij}^{rk} \le b_i \sum_{k\in K} x_{ij}^k \qquad \forall r \in R,\ \forall (i,j) \in A \tag{C.17}$$

$$a_i x_{ij}^k \le T_{ij}^{rk} \le b_i x_{ij}^k \qquad \forall r \in R,\ \forall k \in K,\ \forall (i,j) \in A \tag{C.18}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall k \in K,\ \forall (i,j) \in A \tag{C.19}$$

$$T_{ij}^{rk} \ge 0 \qquad \forall r \in R,\ \forall k \in K,\ \forall (i,j) \in A \tag{C.20}$$

The objective (C.8) sums up the cost of the used edges. Constraints (C.9) ensure that all customers are visited exactly once, while the redundant constraints (C.10) ensure that no customer is visited more than once. Constraints (C.11) maintain flow conservation between the original nodes $V$, and can be rewritten as

$$\sum_{k\in K}\sum_{(j,i)\in\delta^-(i)} x_{ji}^k = \sum_{k\in K}\sum_{(i,j)\in\delta^+(i)} x_{ij}^k \qquad \forall i \in V$$

since $\sum_{k\in K} x_{ii}^{k-1} = \sum_{k\in K} x_{ii}^k$. Constraints (C.12) maintain flow conservation within a layer. Constraint (C.13) ensures that $K$ partial paths are selected and constraints (C.14) that the length of the partial path in each layer is at most $L$. Constraints (C.15) connect the resource variables on a global level and constraints (C.16) connect the resource variables within each single layer, note that since there is no (C.15) and (C.16) for the depot it is not constrained by resources. Constraints (C.17) globally enforce the resource windows and the redundant constraints (C.18) enforce the resource windows within each layer.

## C.4 Dantzig-Wolfe decomposition

The 3-index formulation of the VRPTW (C.8)–(C.20) is Dantzig-Wolfe decomposed whereby a master and a pricing problem is obtained.

**Master problem:** Let $\lambda_p$ be the variable indicating the use of partial path $p$. Using Dantzig-Wolfe decomposition where the constraints (C.9), (C.11), (C.13), (C.15), and (C.17) are kept in the master problem the following master problem is obtained:

$$\min \sum_{p \in P} c_p \lambda_p \tag{C.21}$$

$$\texttt{s.t.} \sum_{p \in P} \sum_{(i,j) \in \delta^+(i)} \alpha_{ij}^p \lambda_p = 1 \qquad \forall i \in C \tag{C.22}$$

$$\sum_{p \in P : e^p = i} \lambda_p = \sum_{p \in P : s^p = i} \lambda_p \qquad \forall i \in V \tag{C.23}$$

$$\sum_{p \in P} \lambda_p = K \tag{C.24}$$

$$\sum_{(j,i) \in \delta^-(i)} \left( T_{ji}^r + \sum_{p \in P} \tau_{ji}^r \alpha_{ji}^p \lambda_p \right) \leq \sum_{(i,j) \in \delta^+(i)} T_{ij}^r \qquad \forall r \in R,\ \forall i \in C \tag{C.25}$$

$$a_i \sum_{p \in P} \alpha_{ij}^p \lambda_p \leq T_{ij}^r \leq b_i \sum_{p \in P} \alpha_{ij}^p \lambda_p \qquad \forall r \in R,\ \forall (i,j) \in A \tag{C.26}$$

$$T_{ij}^r \geq 0 \qquad \forall r \in R,\ \forall (i,j) \in A \tag{C.27}$$

$$\lambda_p \in \{0,1\} \qquad \forall p \in P \tag{C.28}$$

Where $\alpha_{ij}^p$ is the number of times arc $(i,j) \in A$ is used on path $p \in P$ and $s^p$ and $e^p$ indicates the start respectively the end node of partial path $p \in P$. Constraints (C.22) ensure that each customer is visited exactly once. Constraints (C.23) link the partial paths together by flow conservation. Constraint (C.24) is the convexity constraint ensuring that $K$ partial paths are selected. Constraints (C.25) and (C.26) enforce the resource windows.

**Pricing problem:** The $|K|$ pricing problems corresponding to the master problem (C.21)–(C.28) contains constraints (C.10), (C.12), (C.14), (C.16), and (C.18) and can be formulated as a single ESPPRC where the depot is allowed to be visited more than once. Let $s$ and $e$ be a super source

respectively a super target node. Arcs $(s,i)$ and $(i,e)$ for all $i \in V$ are added to $G$.

$$\min \sum_{(i,j)\in A} \bar{c}_{ij} x_{ij} \tag{C.29}$$

$$\text{s.t.} \sum_{(s,i)\in\delta^+(s)} x_{si} = 1 \tag{C.30}$$

$$\sum_{(i,e)\in\delta^-(e)} x_{ie} = 1 \tag{C.31}$$

$$\sum_{(i,j)\in A} x_{ij} \leq 1 \qquad \forall i \in C \tag{C.32}$$

$$\sum_{(j,i)\in\delta^-(i)} x_{ji} = \sum_{(i,j)\in\delta^+(i)} x_{ij} \qquad \forall i \in V \tag{C.33}$$

$$\sum_{i\in C} \sum_{(i,j)\in A} x_{ij} \leq L \tag{C.34}$$

$$\sum_{(j,i)\in\delta^-(i)} (T^r_{ji} + \tau^r_{ji} x_{ji}) \leq \sum_{(i,j)\in\delta^+(i)} T^r_{ij} \qquad \forall r \in R,\ \forall i \in C \tag{C.35}$$

$$a_i x_{ij} \leq T^r_{ij} \leq b_i x_{ij} \qquad \forall r \in R,\ \forall (i,j) \in A \tag{C.36}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{C.37}$$

The objective (C.29) minimizes the reduced cost of a column. Constraints (C.30) and (C.31) ensure that the path starts in $s$ respectively ends in $e$. Constraints (C.32) dictates that no node is visited more than once, thereby ensuring elementarity, and constraints (C.33) conserve the flow. Constraints (C.35) and (C.36) ensure the resource windows are satisfied for all customers. Note, since the depot is missing in (C.35) each time a path leaves the depot a resource is only restricted by its lower limit $a^r_0$ for all $r \in R$.

Let $\pi$ ($\pi_i \geq 0 : \forall i \in C$) be the duals of (C.22), let $\pi_0 = 0$, let $\mu$ be the duals of (C.23), let $\beta \leq 0$ be the dual of (C.24), let $\nu$ ($\nu \leq 0 : \forall i \in C$) be the duals of (C.25), let $\nu_0 = 0$, and let $\underline{\omega} \leq 0$ and $\overline{\omega} \geq 0$ be the dual of (C.26). The cost of the arcs in this ESPPRC are then given as:

$$\bar{c}_{ij} = -\beta + \begin{cases} c_{ij} - \pi_i - \tau_{ij}\nu_j - a_i\underline{\omega}_i + b_i\overline{\omega}_i & \forall (i,j) \in A \setminus (\delta^+(s) \cup \delta^-(e)) \\ \mu_j & \forall (s,j) \in \delta^+(s) \\ \mu_i & \forall (i,e) \in \delta^-(e) \end{cases}$$

and the pricing problem becomes finding the shortest path from $s$ to $e$.

**Solving the pricing problem:** ESPPRCs can be solved by a labeling algorithm. For details regarding labeling algorithms we refer to Desaulniers et al. [4], Irnich [6], Irnich and Desaulniers [7], and Righini and Salani [10].

**Branching:** Integrality can be obtained by branching on the original variables, which can be accomplished by cuts in the master problem (see Vanderbeck [11]), e.g., let $X_{ij}$ be the set of partial paths that utilize arc $(i,j)$ then the branch rule $x_{ij} = 0 \ \vee \ x_{ij} = 1$ can be expressed by:

$$\sum_{p\in X_{ij}} \lambda_p = 0 \ \vee \ \sum_{p\in X_{ij}} \lambda_p = 1.$$

**Bounds:** The following theorem justifies the approach presented in this paper.

**Theorem 1.** *Let $z_{lp}$ be an LP-relaxed solution to* (C.1)–(C.7) *and let $z_{pp}$ be an LP-relaxed solution to* (C.21)–(C.28) *then $Z_{lp} \leq Z_{pp}$ for all instances of VRPTW and $Z_{lp} < Z_{pp}$ for some instances of VRPTW.*

*Proof.* $Z_{lp} \leq Z_{pp}$ since all solutions to (C.21)–(C.28) map to solutions to (C.1)–(C.7). An instance with $Z_{lp} < Z_{pp}$ is obtained with four customers each with a demand of resource $r$ of half the global maximum $b_r$ of $r$, the distance from the customers to the depot larger than the distance between the customers, and $L = 4$. The solution to (C.21)–(C.28) would use the expensive edges four times, whereas the solution to (C.1)–(C.7) only would use them twice. □

## C.5 Conclusion

A new decomposition model of the VRPTW has been presented with ESPPRCs as the pricing problems. The model facilitates control of the running time of the pricing problems. Due to the aggregation of the model, LP relaxed bounds of (C.21)–(C.28) are better than the direct model (C.1)–(C.7). Since (C.21)–(C.28) is a relaxation of the traditional Dantzig-Wolfe decomposition model with elementary routes as columns, the LP relaxed bounds may be weaker yielding a larger branch-and-bound tree. The difference in bound quality can be decreased with the use of special purpose cutting planes, which this paper does not leave room for. Furthermore, effective cuts such as Subset Row-inequalities by Jepsen et al. [8] and Chvátal-Gomory Rank-1 cuts (see Petersen et al. [9]) can be applied to the Set Partition master problem to strengthen the bound. Future experimental results will conclude on the effectiveness of this approach.

# Bibliography


[1] N. Ascheuer, M. Fischetti, and M. Grötschel. Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. *Mathematical Programming*, 90(3):475–506, May 2001.

[2] J. F. Bard, G. Kontoravdis, and G. Yu. A branch-and-cut procedure for the vehicle routing problem with time windows. *Transportation Science*, 36(2):250–269, May 2002.

[3] N. Christofides and S. Eilon. An algorithm for the vehicle-dispatching problem. *Operational Research Quarterly*, 20(3):309–318, 1969.

[4] G. Desaulniers, J. Desrosiers, J. Ioachim, I. M. Solomon, F. Soumis, and D. Villeneuve. A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T. G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, chapter 3, pages 57–93. Springer, 1998.

[5] M. Dror. Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, 42:977–979, 1994.

[6] S. Irnich. Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectrum*, 30(1):113–148, January 2008.

[7] S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, Jacques Desrosiers, and M.M. Solomon, editors, *Column Generation*, chapter 2, pages 33–65. Springer, 2005.

[8] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle routing problem with time windows. *Operations Research*, 56(2):497–511, March–April 2008.

[9] B. Petersen, D. Pisinger, and S. Spoorendonk. Chvátal-Gomory rank-1 cuts used in a Dantzig-Wolfe decomposition of the vehicle routing problem with time windows. In B. Golden, R. Raghavan, and E. Wasil, editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 397–420. Springer, 2008.

[10] G. Righini and M. Salani. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3): 255–273, 2006.

[11] F. Vanderbeck. On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Operations Research*, 48(1):111–128, January–February 2000.

# Bilag D

# Danish Summary

Dette Ph.D projekt giver et overblik over, hvordan de to matematiske løsningsmetoder Branch-and-Cut og Branch-and-Cut-and-Price kan benyttes i forbindelse med løsning af ruteplanlægningsproblemer. Ph.D projektet behandler dels to klassiske ruteplanlægningsproblemer, det relaterede ressource begrænsede korteste vej-problem og et nyt storby ruteplanlægnings problem. I det klassiske rute planlægnings problem skal et sæt af kunder have leveret varer fra et centralt depot. Kundernes varebehov måles i kilo og køretøjet har en grænse for hvor mange kilo det kan køre med. Målet for optimeringsalgoritmerne er at finde det sæt af ruter der har den mindste totale længde. En klassisk udvidelse er at kunder har et tidsvindue dvs. køretøjet skal ankomme mellem for eksempel 8 og 16 for at kunne aflæse varene. For at håndtere de problemer der er i mange storbyer, hvor det ofte enten ikke er tilladt eller er meget besværligt at køre med de helt store lastvogne er et nyt ruteplanlægnings problem kaldet 2ECVRP blevet analyseret. I 2ECVRP er der foruden kunderne og det centralt depot også satellitdepoter som er beliggende i udkanten af byerne. De store køretøjer kører så mellem depotet og satellitdepoter og mindre køretøjer sørger så for at varene kommer fra satellitdepoter til kunderne. Målet for optimerings algoritmerne er igen at finde det sæt af ruter der har den mindste totale længde. Alle algoritmer er implementeret og testet på en moderne computer og i flere tilfælde er der fundet optimale løsninger til problemer som ikke har kunnet løses tidligere.